Diseño de Bases de Datos Basadas en Documento: Modelo de Interrelación de Documentos

Gerardo Rossel, Andrea Manna

Departamento de Computación. FCEyN.
Universidad de Buenos Aires
{grossel,amanna}@dc.uba.ar

Resumen Las bases de datos no relacionales han experimentado en los últimos años un importante crecimiento, particularmente las bases de datos basadas en documento. A partir de la amplia adopción de este tipo de modelo de almacenamiento, aumenta la necesidad de contar con herramientas de modelización adecuadas. En este trabajo presentamos una metodología de modelización conceptual y de diseño de este tipo de bases de datos, para lo cual proponemos además la realización de un tipo de diagrama que denominamos *diagrama de Interrelación de documentos* o *DID*.

Keywords: NoSQL, Bases de datos basadas en documentos, modelización conceptual de datos, diagrama de diseño de documentos, BigData, JSON

1 Introducción

El enorme crecimiento de lo que se ha denominando *Big Data*, entendido como el análisis, procesamiento y almacenamiento de grandes cantidades de datos, que se ha producido en los últimos años ha impactado fuertemente en la tecnología de almacenamiento de datos. Entre los desafíos que se plantean se encuentran: la necesidad de escalar en forma horizontal, el trabajo con fuentes de datos diversas, la falta de esquema o estructura de los datos con los que se trabaja, etc.. Estas demandas, junto a la necesidad de alcance global y disponibilidad permanente, tuvieron como respuesta el surgimiento de una familia de bases de datos, que no se referencian en el modelo relacional, conocidas como NoSQL (en algunos contextos también denominadas *NoSQL datastores*).

Si bien hay una enorme variedad de bases de datos *NoSQL* podemos clasificarlas, entre otras cosas, por la forma de almacenar y recuperar la información[4][5]:

- Clave/Valor (Key-Value).
- Basadas en Documento.
- Familia de columnas.
- Basadas en Grafos.

En este artículo trabajaremos sobre una de las categorías más utilizadas: las bases de datos NoSQL basadas en documentos, una de cuyos exponentes (MongoDB) se ha ubicado en el cuarto lugar de popularidad entre las bases de datos, siendo la más popular entre los sistemas NoSQL[6]. Este tipo de base de datos utilizan un enfoque similar a las bases clave/valor pero con importantes diferencias: los valores son almacenados como documentos en un formato standard como *Extensible Markup Language* (XML) o principalmente como *JavaScript Object Notation* (JSON) y es posible realizar consultas basadas en el contenido de los documentos[5].

Las metodologías tradicionales de diseño y construcción de bases de datos en el mundo relacional han sido ampliamente estudiadas y refinadas por décadas de investigación [1][2][3]. En la Fig. 1 se muestra un flujo simplificado de los pasos que se siguen en el diseño tradicional.

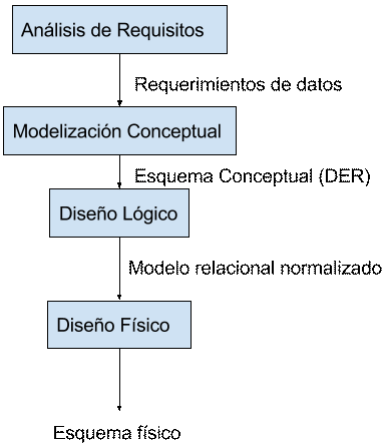


Fig. 1: Pasos generales en el diseño de bases de datos relacionales

La modelización conceptual se presenta como una descripción de los requerimientos respecto de los datos y es realizada usualmente utilizando alguna forma de diagrama de entidad-relación (DER) [2]. En esta fase también se analizan los requerimientos de manipulación de datos de alto nivel que a su vez sirven para la validación del modelo. Posteriormente, se sigue el mapeo del modelo conceptual al modelo relacional y la normalización de éste. Estas tareas forman parte del diseño lógico y tiene aspectos independientes y aspectos dependientes del sistema de gestión base de datos. Los principios de diseño utilizados en el diseño lógico no son aplicables a las bases de datos basadas en documentos. Estas últimas conllevan un conjunto de decisiones basadas en los patrones de consulta que implica muchas veces redundancia y *desnormalización*. En este trabajo proponemos un método de modelización lógica adecuada para este tipo de bases de datos y para ello incorporamos un nuevo tipo de diagrama que llamamos *diagrama de inter-*

relación de documentos o *DID*. Utilizamos la modelización conceptual basada en los diagramas de entidad-relación y sumamos los patrones de consulta que surgen del análisis de requisitos. El resto de este artículo es como sigue: la sección 2 provee un fundamento básico del modelo de documentos, la sección 3 detalla la modelización conceptual y el rol de los patrones de consulta, la sección 4 desarrolla la forma en que a partir del modelo de entidad-relación y los patrones de consulta se construye el modelo de diseño de documentos utilizando el *DID*.

2 Modelo de documentos

La palabra *documento* en el contexto de una base de datos basada en documentos significa una estructura jerárquica organizada de datos. La utilización de *JSON* (o incluso *Binary JSON* o *BSON*) para el almacenamiento de documentos es la forma dominante.

Similar a lo que ocurre con las bases de datos *clave/valor*, se almacenan los documentos en la parte del *valor*. Una de las características comunes en este tipo de base de datos es que son agnósticas respecto al esquema de los documentos. Los documentos similares se agrupan en colecciones, cómo las tuplas en una base de datos relacional se agrupan en tablas. A diferencia de un esquema relacional donde todas las tuplas de una tabla tienen la misma estructura los documentos de una colección pueden diferir en su estructura. Esto permite que a la flexibilidad para almacenar documentos con diferentes estructuras en la misma colección se sume la posibilidad de incorporar cambios en los datos sin necesidad de reestructurar la base de datos. En el listado 1 vemos dos documentos con diferente estructura que pueden ser guardados en la misma colección

Listado 1: Dos documentos con diferente estructura

```
{
"_id": 1
"nombre":"Napoleon",
"apellido":"Solo",
"genero": "masculino",
"fecha_nacimiento":"1960-08-03T12:57:32.7060000",
"Direccion": "Calle 2 CABA ARGENTINA",
"Telefono": "555-5555"
}
{
"_id": 2
"nombre":"Jhon",
"apellido":"Doe",
"genero" :"masculino",
"fecha_nacimiento":"2003-08-03T12:57:32.7060000",
"Direccion": "Calle 1 CABA ARGENTINA",
"idiomas":["ingles","español"]
}
```

Lo que en principio parece sencillo se complica cuando se debe decidir la forma de relacionar documentos. A diferencia del modelo relacional, no es posible realizar una junta como consulta. Más adelante detallamos los diferentes patrones para modelar la relación entre documentos.

3 Modelización conceptual y diseño de documentos

Nuestra propuesta comienza por la realización de un modelo conceptual de los datos que nos permite obtener conocimiento de los datos que deben administrarse. Para ello optamos por la utilización del diagrama de entidad-relación usando la notación de Chen [2] que además de ser ampliamente conocido en la comunidad de base de datos es una notación independiente de la tecnología e incluso del modelo relacional [7].

La modelización conceptual vinculada al diseño de bases de datos NoSQL es un área que está en desarrollo. Como antecedentes podemos citar la metodología para Apache Cassandra propuesta en [7] que si bien utiliza el DER, está orientada a ese motor específicamente por lo que no es genérica. Otra propuesta sobre modelo conceptual y diseño de NoSQL es la descrita en [10] donde se plantea utilizar los aspectos comunes de las diversas bases NoSQL para realizar una metodología general, donde la modelización conceptual utiliza un modelo abstracto de datos denominado NoAM.

El modelo conceptual en el diseño tradicional es mapeado a un esquema de implementación que se expresa, en general, como un conjunto de esquemas relacionales. En nuestro caso el modelo conceptual debe ser mapeado a un modelo de diseño de documentos. En dicho modelo se representa la estructura general de los documentos en base a colecciones. Para realizar el modelo de diseño de documentos incorporamos un nuevo tipo de diagrama que denominamos diagrama de diseño de documentos y se representa como un extensión del diagrama de entidad-relación al que se le agregan las relaciones propias entre documentos.

Básicamente hay dos formas de relacionar documentos: referenciar o embeber (anidar). La facilidad de embeber documentos permite al diseñador almacenar datos relacionados como un documento simple. De esta manera, se puede romper con lo que se denomina *impedance mismatch*, es decir con la diferencia entre las estructuras de datos en memoria y la forma en que éstos son almacenados [5]. La decisión de cuando embeber o referenciar es una decisión de diseño que debe ser guiada por la forma en que los datos son manipulados. Datos que se acceden regularmente juntos deberían agruparse.

Para poder realizar el diagrama de diseño de documentos a partir del DER, adecuadamente, es necesario ampliar nuestro modelo conceptual con los patrones de consulta y acceso a los datos. Ambas cosas son esenciales y no deben ser subestimadas. Un diseño conceptual equivocado o patrones de consulta errados llevan a un mal diseño de la base de datos. Los patrones de consulta pueden ser escritos en lenguaje informal o en un lenguaje más formal cómo ERQL [8]. Junto a la especificación de las consultas se debe establecer el peso de éstas en el global. Es decir, no es lo mismo una consulta que se realiza esporádicamente que una

que es realizada con mayor frecuencia. Además, es necesario determinar cuales son las operaciones de actualización/inserción más frecuentes y si la aplicación será de lectura intensiva o no. El análisis transaccional puede ayudar también ya que en este tipo de base de datos la unidad atómica de almacenamiento es el documento. Por último el ciclo de vida del documento pasa a ser de interés en un diseño optimo. Conocerlo nos ayuda a reservar espacio en determinados casos como cuando se almacenan series de tiempo.

El diagrama de la Fig.2 muestra un esquema simplificado de como se modifican los pasos del diseño tradicional .

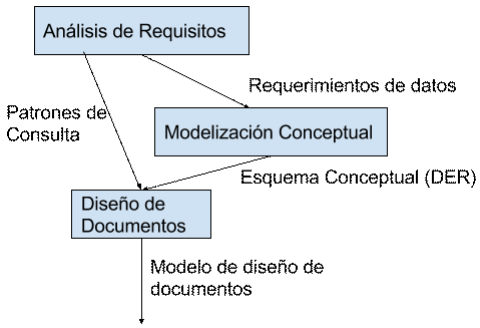


Fig. 2: Pasos en la modelización de documentos

Analizaremos, caso por caso, como llevar el DER al modelo de diseño de documentos en función de los tipos de interrelaciones, su cardinalidad y tomando los patrones de consultas como herramienta para decidir. De este diseño surgen las colecciones que finalmente integrarán la base de datos.

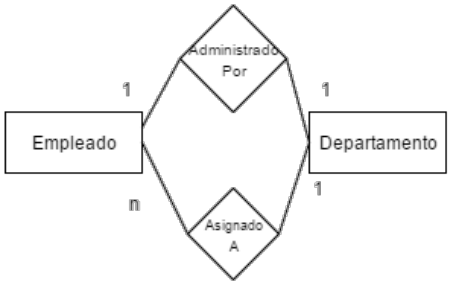
4 Modelo de Diseño de Documentos

En esta sección describiremos como realizar el modelo de diseño de do-cumentos a partir del DER. Algunos de los patrones que seguiremos son conocidos pero en nuestro caso la decisión de diseño queda expresada en el *DID*. El alcance del artículo no permite una descripción exhaustiva de todos los casos por lo que intentaremos mostrar la semántica y sintaxis del *DID* más allá de la cantidad de variantes de diseño que pueden encontrarse.

4.1 Interrelaciones 1 a 1

En la Fig. 5a vemos un DER, tomado del libro de Jeffrey D. Ullman[9]. Se modelan como entidades separadas los empleados y los gerentes (se omiten los atributos por claridad). En este caso es muy posible que, a menos que los patrones de consulta indiquen otra cosa, lo correcto sea embeber un documento en otro y

(a) Gerente y Empleado como entidades separadas



(b) Gerente modelado por la interrelación

Fig. 3: DER Interrelaciones 1 a 1

como además los departamentos tienen otra interrelación lo ideal es embeber al gerente en el departamento. En la Fig. 5b vemos otro DER que modela a los gerentes como empleados diferenciados por participar de la relación *"administrado por"*.

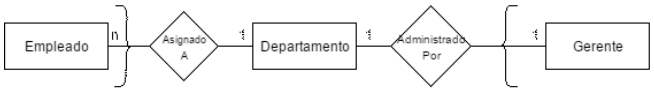
En definitiva, tenemos varios modelos posibles para el diseño de documentos. La notación utilizada en *DID* mantiene al DER pero las entidades pasan a ser documentos e indicamos la forma de relacionarse. Para el caso de embeber, usamos un símbolo específico en forma de llave que lo indica. La Fig. 4 muestra que el documento *Gerente* es embebido en *Departamento*.

Fig. 4: DID. 1 a 1 embebiendo

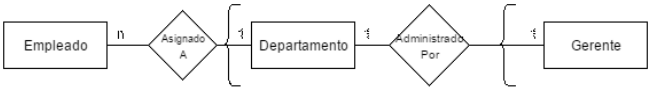
4.2 Interrelaciones 1 a N

En las interrelaciones 1 a N los elementos de una entidad E_1 se interrelacionan con uno o más elementos de otra entidad E_2, mientras que los elementos de E_2 sólo se interrelacionan a un elemento E_1. En la Fig. 3 vemos que un departamento tiene muchos (N) empleados asignados y un empleado puede ser asignado a

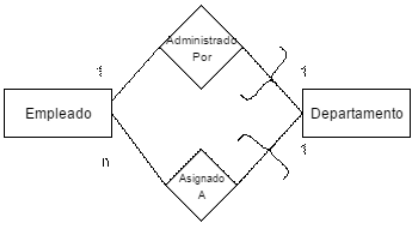
sólo un (1) departamento. También podría haber participación parcial, es decir que existan departamentos que no tienen empleados asignados o que existan empleados que no tengan departamento asignado (esto se indica con una O sobre la línea que vincula la entidad correspondiente con la interrelación).



(a) Empleados Embebidos en Departamento



(b) Departamento embebido en Empleado



(c)

Fig. 5: DID Interrelaciones 1 a N Embebiendo

El diseño de este tipo de interrelación tiene algunas variantes. Un diseño implicaría embeber el documento que representa a E_1 en el documento que representa a E_2 (en el ejemplo sería embeber el documento de Departamento en el documento de Empleado). También es posible que el departamento tenga una lista de documentos que representan Empleados ¿ Cuál solución es mejor?. Dependerá del patrón de consultas. La Fig. 5 nos muestra el *DID* con diferentes formas de embeber. Además de embeber se puede referenciar y también es posible combinar ambas. En la Fig. 6 vemos las referencias indicadas como una flecha abierta. En este caso, los departamentos tendrán una lista de referencias a sus empleados y cada documento empleado una referencia a un departamento. La cardinalidad de la interrelación indica la existencia de una lista o no.

Fig. 6: DID. Referencias

4.3 Interrelaciones N a M

En las interrelaciones N a M (muchos a muchos) los elementos de una entidad E_1 se interrelacionan con uno o más elementos de otra entidad E_2 y viceversa. Nuevamente puede también existir una participación parcial u opcional. En la Fig. 7, se muestra el DID (7b) correspondiente a un DER con una interrelación N a M (7a).

(a) DER con interrelación N a M

(b) DID con N a M usando referencias

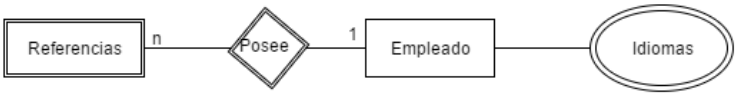
Fig. 7: DID Interrelaciones M a N

4.4 Entidades débiles y atributos multivaluados

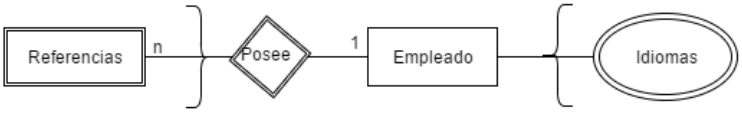
El caso de los atributos multivaluados, aquellos atributos que pueden tener muchos valores para el mismo elemento, y las entidades débiles, que no contienen una clave por si mismas y dependen de otra entidad para ser identificadas, puede ser modelado embebiendo como muestra en las figuras 8a y 8b

4.5 Desnormalización

La normalización es una técnica de diseño utilizada ampliamente en las bases de datos relacionales. La normalización permite obtener esquemas de relación que minimizan la redundancia y evitan anomalías en la actualización e inserción [9]. El costo de la normalización es la necesidad de utilizar juntas para recuperar la información. En los modelos de documentos se utiliza lo que se denomina desnormalización que implica agregar redundancia para optimizar la lectura a costa de trasladar al programador la responsabilidad de evitar inconsistencias. Hasta aquí hemos mostrado cómo se representan en el modelo de documentos la desnormalización más trivial que es cuando un documento se embebe en otro. Una forma menos trivial y que depende fuertemente del patrón de consultas es embeber sólo algunos atributos de un documento en otro. Es necesario que el



(a) DER. Entidades Débiles y Atributos Multivaluados

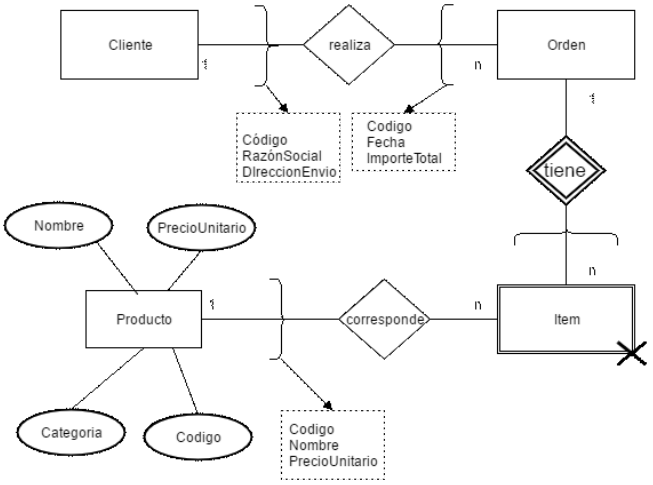


(b) DID. Entidades Débiles y Atributos Multivaluados

Fig. 8: Entidades Débiles y Atributos multivaluados

diagrama permita modelar esta situación lo cual se logra asignando una lista de campos a embeber en el símbolo correspondiente.

Al embeber un documento completo debe decidirse si el documento que se embebe tendrá además una existencia independiente. Esa decisión se indica en el modelo de la siguiente manera: para indicar que un documento ***sólo*** se almacena embebido y no en forma independiente, se marca con una cruz en el diagrama. En la Fig. 9a se puede ver un diagrama donde se muestra como modelar lo descrito.



(a) Denormalización

Se indica que el documento *Item* solamente será almacenado como embebido en el documento *Orden* y que además contendrá sólo el código, el nombre y el precio unitario del producto. Se ven dos embebidos parciales más, el *Cliente*

mantendrá un vector de ordenes con únicamente el código, el precio y el importe total, mientras que la *Orden* sólo mantendrá el código, la razón social y la dirección de envío del cliente. La figura omite los atributos para facilitar la lectura.

5 Conclusiones

En este artículo introdujimos una nueva forma de llevar a cabo el diseño de bases de datos NoSQL basadas en documentos. El enfoque modifica el método tradicional incorporando los patrones de consulta y nuevo modelo de diseño expresado en un diagrama de diseño de documentos. Hemos mostrado como representar en el diagrama los diferentes esquemas de diseño que pueden realizarse al construir una base de datos basada en documentos.

Se observan dos aspectos destacables:

- Es el primer trabajo sobre modelización conceptual y modelización de documentos utilizando un diagrama específico para modelar documentos.
- La simplicidad del diagrama presentado facilita la comunicación en el grupo de desarrollo y la discusión sobre las decisiones de diseño que se tomen.

En el futuro pretendemos refinar este trabajo a partir de la experiencia de su uso en diversos proyectos de desarrollo.

References

1. E. F. Codd. A relational model of data for large shared data banks. Commun. ACM 13, 6 1970, 377-387
2. Peter P. S. Chen. 1975. The entity-relationship model: toward a unified view of data. In Proceedings of the 1st International Conference on Very Large Data Bases (VLDB '75). ACM, New York, NY, USA
3. Peter P. Chen. 2009. Thirty Years of ER Conferences: Milestones, Achievements, and Future Directions. In Proceedings of the 28th International Conference on Conceptual Modeling (ER '09), Alberto H. Laender, Silvana Castano, Umeshwar Dayal, Fabio Casati, and José Palazzo Oliveira (Eds.). Springer-Verlag, Berlin, Heidelberg
4. Adam Flowler: The State of NoSQL 2016. Adam Fowler; 1 edition(2016)
5. Pramod J. Sadalage, Martin Fowler. 2012. NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence (1st ed.). Addison-Wesley Professional.
6. solid IT, DB-Engines Ranking `http://db-engines.com/en/ranking`
7. Artem Chebotko, Andrey Kashlev, Shiyong Lu, A Big Data Modeling Methodology for Apache Cassandra, IEEE International Congress on Big Data (BigData'15), pp. 238-245, New York, USA, 2015.
8. M. Lawley, R. W. Topor. A query language for EER schemas in Proceedings of the 5th Australasian Database Conference, 1994, pp. 292–304.
9. Jeffrey D. Ullman. . Principles of Database and Knowledge-Base Systems, Vol. I. Computer Science Press, Inc., New York, NY, USA. 1988
10. Francesca Bugiotti, Luca Cabibbo, Paolo Atzeni, Riccardo Torlone. Database Design for NoSQL Systems. International Conference on Conceptual Modeling, Oct 2014, Atlanta, United States. pp.223 - 231, 2014.