Diseño y comparación de performance de una base de datos relacional SQL y una base de datos documental en MongoDB

Sebastián Peláez
Ingeniería en Computación
Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay
pelaez.seba@gmail

Joaqun Torres
Ingeniería en Computación
Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay
joatorresyic@gmail.com

***Resumen*—El objetivo es trabajar a partir de una base de datos relacional SQLServer (con datos reales), diseñar una base de datos documental en MongoDB optimizando un conjunto de consultas problemáticas. Se realizan pruebas de desempeño sobre ambos modelos usando una tabla de personas con 500.000 registros y una tabla de auditoría con 1.500.000 registros. Analizados los resultados se concluye que para las consultas problemáticas en el modelo relacional, sobre el nuevo modelo documental de la tabla de auditoría, el tiempo se redujo de 65 segundos a 7 segundos.**

I. Introducción

A partir de un problema planteado por una empresa de Software del Área de la Salud, que administra una base de datos SQLServer[1] con un gran volumen de datos, surge la necesidad de mejorar el tiempo de ejecución de consultas frecuentes.

Se cuenta con una tabla maestra de personas (IOPERSONAS) que contiene el registro de todos los usuarios que han accedido a alguno de sus sistemas, ya sea como usuarios, como administradores o con otro rol. De algunos de estos sistemas, surge la necesidad de hacer consultas de búsqueda sobre las personas, filtrando por primer nombre o primer apellido. Se plantea comparar el desempeño del modelo actual en SQLServer y el nuevo modelo diseñado en una base de datos documental en MongoDB[2], utilizando un conjunto de consultas seleccionado.

Una importante cantidad de sistemas escriben sobre una tabla de Auditoría (TAULOG), en donde se almacena el registro de las acciones que realiza un usuario. Por las características de la misma, y al ser usada por sistemas muy diversos, el registro de acciones se guarda en una columna de la tabla de tipo string pero que se escribe como JSON. El personal que trabaja en el análisis de datos, necesita realizar consultas sobre los campos del JSON. El objetivo es diseñar una base de datos Documental en MongoDB, a partir de los datos de la tabla SQLServer, convirtiendo el JSON en un campo del documento. Se realiza una comparación del desempeño de ambas bases de datos, para las consultas más frecuentes.

El documento se encuentra organizado de la siguiente forma; en la sección II se muestran los trabajos relacionados con la temática de este informe. En la sección III se describe el diseño de las bases de datos con las que se trabaja y las decisiones que se toman para implementarlas, en particular al pasar del modelo relacional al documental. En la sección IV se detallan las características de los datos de prueba y de donde se obtienen, el ambiente en el que se realizan las pruebas, las consultas realizadas para las mismas y se exponen los resultados. Por último, en la sección IV se concluye trabajo realizado y se sugieren futuras tareas para continuar la mejora de este proyecto.

II. Trabajos relacionados

En los siguientes artículos se compara el desempeño entre bases de datos en distintos modelos.

En el trabajo de [Veronika Abramova(2014)][3] se comparan dos bases de datos documentales, presentando los datos y resultados que surgen de esa comparación, luego se utiliza como referencia para comparar la base de datos relacional SQLServer y la base de datos en MongoDB.

En el trabajo de [Ciprian-Octavian Truica(2018)][4], se realiza una comparativa de desempeño de las operaciones CRUD para distintas bases de datos, tanto relacionales como no relacionales. En este informe se busca analizar en un ambiente con replicaciónn de datos asincrónicos.

[1] https://www.microsoft.com/es-es/sql-server/sql-server-2019
[2] https://www.mongodb.com/
[3] http://www.ronpub.com/journals/ojdb
[4] https://arxiv.org/pdf/1806.04761.pdf

III. Diseño e Implementación

III-A. Tabla IOPERSONAS en base de datos relacional

La tabla IOPERSONAS en el modelo relacional se representa con una clave primaria y 36 columnas que hacen referencia a la información de las personas.

III-B. Colección IOPERSONAS en la base de datos documental

A partir de la tabla relacional se diseña la colección de personas para el modelado en documental. Se decide mantener todos los campos representados dentro de una única colección en donde cada uno de ellos se corresponde con las columnas de la tabla de personas del modelo relacional en SQLServer.

III-C. Tabla TAULOG en base de datos relacional

La tabla TAULOG en el modelo relacional actual se representa con una clave primaria y 12 columnas de información que hacen referencia al usuario y el sistema en que está realizando la auditoría.

Se pretende centrar la atención en el campo AULOGMETDAT que es un string donde se almacena la información de valor para la auditoría. En algunos casos, puede contener un mensaje de error o éxito, en cambio en otros, se almacenan objetos JSON convertidos a string (entradas y salidas en el llamado a una rutina o servicio).

III-D. Colección TAULOG en la base de datos documental

Para el modelado de ésta colección en documental se tienen en cuenta las consultas para las que se busca mejorar el desempeño, todas ellas sobre el campo AULOGMETDAT. Se desarrolla un script que recorre toda la tabla en busca de strings que representan un JSON para convertirlo a un objeto de ese tipo. En caso de falla, de ser posible, se crea una expresión regular para recuperar el objeto. Las fallas más comunes resultan de a caracteres incorrectos o de escape que no hace posible la conversión. En caso de no poder recuperar el objeto, ya que por ejemplo, la cadena de caracteres se corta bruscamente debido al exceso en el tamaño del campo que se quiere guardar en la columna de la tabla, se almacena como un string.

A partir del análisis realizado y de la información provista por el cliente, se concluye que el JSON que se almacena tiene una estructura muy heterogénea, dependiendo del sistema que está auditando, es la estructura que tendrá el objeto almacenado. Se resuelve almacenar en el campo AULOGMETDAT utilizando la estructura del JSON que se recibe convertido en una colección de MongoDB. En caso de que lo recibido no sea un objeto JSON o que sea un objeto mal formado, se guarda el contenido como un string.

El resto de los atributos se almacenan de igual forma que en la tabla relacional. La colección se modela con el documento JSON presentado en el Código 1.

Código 1 - Colección de TAULOG

```
{
    "_id": ,
    "AULOGID": ,
    "AULOGFECHOR": ,
    "AULOGUSU": ,
    "AULOGEMPCON": ,
    "AULOGOPE": ,
    "AUOBJID": ,
    "AULOGIP": ,
    "AULOGNAV": ,
    "AULOGOS": ,
    "AULOGREGCLA": ,
    "AULOGMETDAT": ARRAY,
    "AULOGAPPID": ,
    "AULOGINLOGID":
}
```

Índices:

- AULOGID
- AULOGINLOGID
- AULOGREGCLA
- AUOBJID

Para las pruebas es de interés realizar consultas sobre un campo de la colección guardada en AULOGMETDAT, y que en su estructura tenga también una colección (*Val*). Uno de los elementos también es un JSON (*wsRstUpdOrdSrvEnt*), que a su vez tiene una colección (*Datos*), y sobre esos datos poder consultar sobre el atributo *wsRstUpdOrdSrvDatCod*, es por esto se agrega un índice por *AULOGMETDAT.Val.wsRstUpdOrdSrvEnt.Datos. wsRstUpdOrdSrvDatCod*. Se compara el rendimiento antes de agregar el índice y después de agregarlo.

IV. Experimentación

Para realizar las pruebas, se consideran las consultas mas frecuentes que se suelen hacer sobre las tablas del modelo relacional en el sistema que se encuentra en producción y algunas identificadas como poco performantes, de forma de poder comparar el rendimiento de la base relacional SQLServer y la base documental MongoDB. Allí se encuentran consultas sobre diferentes campos, sin índices y con índices, tanto simples como compuestos. Además, se decide realizar pruebas de inserción con varios usuarios concurrentes.

Todas las pruebas se realizan usando la herramienta JMeter[5] la cual nos permite crear un plan de pruebas y cargar cada una

[5]https://jmeter.apache.org

de las consultas para cada una de las pruebas. Luego en los resultados, se muestra un conjunto de métricas que permiten analizar el rendimiento de las consultas.

IV-A. Datos de prueba

Los datos de prueba se obtienen de la base de datos de desarrollo del sistema. Se migran a un ambiente local las tablas IOPERSONAS en donde cada registro contiene los datos de una persona y la tabla en total posee 536.986 registros. Se migra también la tabla TAULOG que posee 1.569.564 registros.

Los datos de la base de MongoDB son los mismos que los de la base SQLServer. Para migrar los datos de ambas tablas se realiza un script en el lenguaje Python que recorre los registros de las tablas del modelado relacional, genera un documento con el nuevo diseño, realiza las transformaciones correspondientes e inserta el mismo en la base documental.

La tabla IOPERSONAS, no implica mayores inconvenientes en la migración de datos, ni en la conversión de los mismos.

La tabla de TAULOG requiere un análisis más detallado por la particularidad de de ser de auditoría. Se encuentran 300.000 registros AULOGMETDAT que eran objetos JSON mal formados, es decir, tenían inconsistencias en su estructura, que hacía que al convertir el string en un objeto JSON ocurriera un error en la ejecución. Se aplican algoritmos para la recuperación del objeto JSON, logrando recuperar la mayoría de los registros, aún así, 6.000 registros que no se pudieron recuperar y se almacenan como strings.

Dado que el proyecto está enmarcado en un sistema del área de la Salud, los datos son sensibles. Por lo tanto, se debió firmar un acuerdo de confidencialidad y no es posible publicar el conjunto de datos en su totalidad.

IV-B. Consultas de prueba

En esta sección se exponen las consultas realizadas para las pruebas de desempeño sobre cada una de las bases de datos.

Las dos primeras consultas corresponden a la búsqueda de una persona por el atributo de nombre (*IOPERNOM1*) sobre el cual no hay ningún índice. La consulta 1 es sobre la base SQLServer y la consulta 2 sobre la base MongoDB.

Consulta 1 Select en tabla personas SQLServer

```
SELECT * FROM IOPERSONAS WHERE IOPERNOM1 like 'CARLOS'
```

Consulta 2 Find en colección personas MongoDB

```
collection.find("IOPERNOM1" : "CARLOS")
```

En las consultas 3 y 4 se busca a una persona por nombre (*IOPERNOM1*) y apellido (*IOPERAPE1*). En este caso, tanto la base SQLServer como la base MongoDB, tienen definido un índice compuesto con esos dos atributos, por lo que estas consultas son de utilidad para medir el rendimiento de los mismos.

Consulta 3 Select en tabla personas SQLServer

```
select * from IOPERSONAS where IOPERNOM1 like 'CARLOS' AND
IOPERAPE1 like 'GARCIA'
```

Consulta 4 Find en colección personas MongoDB

```
collection.find({"IOPERNOM1":"CARLOS","IOPERAPE1":"GARCIA"})
```

La consulta 5 corresponde a la búsqueda en la tabla TAULOG, sobre uno de los campos del JSON almacenado como string en la columna AULOGMETDAT de la tabla relacional. Luego, en la consulta 6 se realiza la búsqueda sobre el mismo dato almacenado en la colección TAULOG de la base en MongoDB. La consulta 7 es la misma consulta que la anterior, pero antes de la ejecución se define un índice sobre el campo que se consulta para poder evaluar las diferencias de tiempo con y sin índice.

Consulta 5 Select en auditoría SQLServer

```
select * from TAULOG
where AULOGMETDAT like '%"wsRstUpdOrdSrvOrd":"18322028"%'
```

Consulta 6 Find en auditoría MongoDB

```
db.TAULOG.find(
{"AULOGMETDAT.Val.wsRstUpdOrdSrvEnt.wsRstUpdOrdSrvOrd"
:"18322028"}
)
```

Consulta 7 Find en auditoría MongoDB

```
db.TAULOG.find(
{"AULOGMETDAT.Val.wsRstUpdOrdSrvEnt.wsRstUpdOrdSrvOrd"
:"18322028"}
)
```

Por las características ya mencionadas de la tabla TAULOG, varios usuarios pueden insertar elementos en ella al mismo tiempo. Para esto se definen los siguientes inserts.

Consulta 8 Insert en tabla TAULOG SQLServer

```
INSERT INTO TAULOG
(AULOGID, AULOGFECHOR, AULOGUSU, AULOGEMPCON, AULOGOPE,
AUOBJID, AULOGIP, AULOGNAV, AULOGOS, AULOGREGCLA,
AULOGMETDAT, AULOGAPPID,AULOGINLOGID)
VALUES((SELECT max(AULOGID) FROM TAULOG)+1,
N'2021-07-17-12.11.28.000000', N'PLATAFORMA', N'"2"',
N'I', 199, N'"192.168.201.300"', 0,
N'RPT-HTTPClient/0.3-3I', N'', N'[{"Nom":"in","Val":
"{"wsRstUpdOrdSrvEnt":{"wsRstUpdOrdSrvOrd":"18322301"
,"Datos":[{"wsRstUpdOrdSrvDatCod":"TELEMEDICINA",
"wsRstUpdOrdSrvDatVal":"+094"}]}}","Tip":"S"},
{"Nom":"out","Val":"{"wsRstUpdOrdSrvSal":
{"wsRstUpdOrdSrvSalStsCod":0,"wsRstUpdOrdSrvSalStsMsg"
:"Via de comunicacion modificada"}}","Tip":"S"},
{"Nom":"method","Val":"POST","Tip":"S"},
{"Nom":"soapaction","Val":"","Tip":"S"},
{"Nom":"duration","Val":"63","Tip":"S"}]', 4, 26);
```

Consulta 9 Insert en colección TAULOG MongoDB

```
db.TAULOG.insertOne( { AULOGID: (db.TAULOG.find().sort(
{AULOGID:-1}).limit(1) +1), AULOGFECHOR:
"2021-07-17-12.11.28.000000", AULOGUSU: "seba",
AULOGEMPCON : '"2"', AULOGOPE: 'I', AUOBJID : 199
, AULOGIP: '"192.168.200.300"', AULOGNAV:0, AULOGOS:
'RPT-HTTPClient/0.3-3I', AULOGREGCLA:'', AULOGMETDAT:
'[{"Nom":"in","Val":"{"wsRstUpdOrdSrvEnt":
{"wsRstUpdOrdSrvOrd":"18322300","Datos":[
{"wsRstUpdOrdSrvDatCod":"TELEMED_TELCON",
"wsRstUpdOrdSrvDatVal":"+094"}]}}","Tip":"S"},
{"Nom":"out","Val":"{"wsRstUpdOrdSrvSal":
{"wsRstUpdOrdSrvSalStsCod":0,
"wsRstUpdOrdSrvSalStsMsg":
"Via de comunicacion modificada"}}","Tip":"S"}
,{"Nom":"method","Val":"POST","Tip":"S"},{"Nom"
:"soapaction","Val":"","Tip":"S"},{"Nom":
"duration","Val":"63","Tip":"S"}]', AULOGAPPID:
4, AULOGINLOGID: 26} );
)
```

IV-C. Configuración del ambiente de pruebas

Las pruebas se realizan en una máquna local, con las siguientes características:

- Sistema operativo: Windows 10 Home 64 bits.
- Procesador: Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz.
- Memoria RAM: 8 GB.
- Disco duro: 512 GB.

En la máquina se configura el ambiente para las pruebas:

- JMeter versión: 5.4.1.
- SQLServer 2014.
- MongoDB server versión: 4.4.5
- Python versión: 3.8.5

IV-C1. Pruebas con JMeter: Para pruebas de desempeño se utiliza la herramienta JMeter. En los siguientes cuadros comparativos se presentan los resultados obtenidos de la ejecución de las pruebas de rendimiento de las tablas de persona y auditoría sobre SQLServer y MongoDB. Para los mismos se muestra cada una de las consultas realizadas; la base de datos en donde se ejecuta; la cantidad de veces que se realiza la consulta; el tiempo promedio, mínimo y máximo que tomo realizar la consulta expresado en milisegundos.

Las consultas de la 1 a la 7 se realizan con un usuario concurrente ya que para ellas en el caso de uso no es necesario evaluar el rendimiento con esa variable. En las consultas 8 y 9 donde se realizan los insert se simulan 100 usuarios concurrentes para acercarse a la realidad planteada.

En todos los casos, previamente a realizar las pruebas se vacía la caché de los motores de base de datos de forma que no altere el resultado.

Cuadro I: Consulta por nombre de persona SQLServer vs MongoDB

Cons	***BD***	***Cant***	***Prom***	***Min***	***Max***
1	SQL	5	1016	931	1093
2	Mongo	5	1562	1373	1863

Cuadro II: Consultas por nombre y apellido de persona SQL-Server vs MongoDB

Cons	***BD***	***Cant***	***Prom***	***Min***	***Max***
3	SQL	5	241	178	381
4	Mongo	5	188	146	234

Cuadro III: Consultas por campo wsRstUpdOrdSrvOrd, SQL-Server vs MongoDB

Cons	***BD***	***Cant***	***Prom***	***Min***	***Max***
5	SQL	3	65613	61552	73096
6	Mongo	3	7633	7215	8414
7	Mongo	5	106	37	247

Cuadro IV: Consultas de inserción de datos SQLServer vs MongoDB

Cons	***BD***	***Cant***	***Prom***	***Min***	***Max***
8	SQL	100*	4052	1309	5704
9	Mongo	100*	2820	1700	4591

* Las 100 ejecuciones se corresponden a los hilos de ejecuciones concurrentes

V. Conclusiones y trabajo futuro

La implementación de la colección TAULOG en la base de datos en MongoDB supera ampliamente el desempeño de la tabla actual TAULOG en la base de datos relacional.

Como se observa en el Cuadro III la principal ventaja en el desempeño se da sobre las consultas a campos del JSON almacenado como colección en el campo AULOGMETDAT, y que en el modelo SQL es un campo del tipo string. Para hacer una búsqueda por like, el motor de base de datos SQLServer necesita recorrer toda la tabla. Para resolver la misma consulta en MongoDB, al ser una colección del documento, es más eficiente por ser un acceso directo. Además, en la base de datos documental, para algunas consultas más usadas, es posible agregar un índice sobre los campos del objeto JSON del campo AULOGMETDAT para lograr un desempeñó aún mejor.

Se puede afirmar que es ampliamente recomendable utilizar una base de datos documental en MongoDB para la tabla de auditorías TAULOG.

Al observar los resultados obtenidos en las pruebas realizadas sobre IOPERSONAS expuestos en los cuadros I y II, no se observan diferencias significativas de tiempos entre la base de datos documental y la base de datos relacional. Por lo tanto, no creemos que merezca el esfuerzo realizar la migración de un modelo a otro.

Los problemas encontrados, como se menciona en la etapa de Diseño e Implementación, están ligados a la migración de datos del campo AULOGMETDAT del modelo relacional y convertirlo a JSON para insertarlo como colección en el modelo documental. Aquí se encuentran objetos mal formados o con errores de sintaxis, que al estar almacenados como string no generaban ningún error, pero al ser convertidos a JSON ocurría un error. Teniendo en cuenta que para las pruebas se utiliza el conjunto de datos del ambiente de desarrollo, se considera necesario hacer un análisis exhaustivo de los JSON que se encuentran almacenados en el ambiente de producción antes de realizar la migración de los datos en dicho ambiente. Para luego ajustar los scripts de transformación y recuperar la mayor cantidad posible de objetos reduciendo al mínimo los que permanezcan almacenados como strings.

Como trabajo a futuro se propone coordinar con todos los sistemas que insertan registros sobre la tabla TAULOG para que se verifique que los objetos JSON insertados en el campo AULOGMETDAT tengan una estructura válida.

REFERENCIAS

[Ciprian-Octavian Truica(2018)] Alexandru Boicea Ion Bucur Ciprian-Octavian Truica, Florin Radulescu. Performance evaluation for CRUD operations in asynchronously replicated document oriented database, 2018.

[Veronika Abramova(2014)] Pedro Furtado Veronika Abramova, Jorge Bernardino. Which NoSQL Database? A Performance Overview, 2014. URL http://www.ronpub.com/journals/ojdb.