



**EDUCACIÓN**

SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO  
NACIONAL DE MÉXICO

Instituto Tecnológico de Orizaba

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

OPCIÓN I.- TESIS

TRABAJO PROFESIONAL

“DESARROLLO DE UN MÉTODO DE  
FRAGMENTACIÓN HÍBRIDA DINÁMICA  
PARA BASES DE DATOS MULTIMEDIA”

QUE PARA OBTENER EL GRADO DE:  
MAESTRO EN SISTEMAS  
COMPUTACIONALES

PRESENTA:

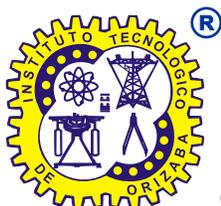
*I.S.C. Oscar Crescencio Rico*

DIRECTOR DE TESIS:

*Dra. Lisbeth Rodríguez Mazahua*

CODIRECTOR DE TESIS:

*M.S.C. Felipe Castro Medina*



ORIZABA, VERACRUZ, MÉXICO.

FEBRERO 2024



Orizaba, Veracruz, **21/febrero/2024**  
Dependencia: **División de Estudios de  
Posgrado e Investigación**  
Asunto: **Autorización de Impresión**  
OPCION: I

**C. OSCAR CRESCENCIO RICO  
CANDIDATO A GRADO DE MAESTRO EN:  
SISTEMAS COMPUTACIONALES  
P R E S E N T E.-**

De acuerdo con el Reglamento de Titulación vigente de los Centros de Enseñanza Técnica Superior, dependiente de la Dirección General de Institutos Tecnológicos de la Secretaría de Educación Pública y habiendo cumplido con todas las indicaciones que la Comisión Revisora le hizo respecto a su Trabajo Profesional titulado:

**" Desarrollo de un método de fragmentación híbrida dinámica para bases de datos multimedia "**

comunico a Usted que este Departamento concede su autorización para que proceda a la impresión del mismo.

**ATENTAMENTE**

*Excelencia en Educación Tecnológica®*  
CIENCIA - TÉCNICA - CULTURA®

**DRA. OFELIA LANDETA ESCAMILLA  
ENCARGADA DE LA DIVISIÓN DE ESTUDIOS  
DE POSGRADO E INVESTIGACIÓN**

OG-13-F06





Orizaba, Veracruz, 12/febrero/2024  
Asunto: Revisión de trabajo escrito

**C. OFELIA LANDETA ESCAMILLA  
JEFE DE LA DIVISIÓN DE ESTUDIOS  
DE POSGRADO E INVESTIGACIÓN  
P R E S E N T E.-**

Los que suscriben, miembros del jurado, han realizado la revisión de la Tesis del (la) C.

**OSCAR CRESCENCIO RICO**

La cual lleva el título de:

**Desarrollo de un método de fragmentación híbrida dinámica para bases de datos multimedia**

Y concluyen que se acepta.

**ATENTAMENTE**  
Excelencia en Educación Tecnológica®  
CIENCIA - TÉCNICA - CULTURA®

**PRESIDENTE: DRA. LISBETH RODRÍGUEZ  
MAZAHUA**

FIRMA

**SECRETARIO: DR. GINER ALOR HERNÁNDEZ**

FIRMA

**VOCAL: DR. JOSÉ LUIS SÁNCHEZ  
CERVANTES**

FIRMA

**VOCAL SUP.: M.S.C. FELIPE CASTRO MEDINA**

FIRMA

TA-09-18



## **Agradecimientos**

Agradezco a Dios por brindarme salud e iluminarme en todo momento, en lo personal y académico, permitiéndome llegar hasta aquí. Así mismo, agradezco a mis padres, hermana y abuelos porque siempre confiaron en mí, me dieron palabras de aliento en los momentos difíciles para seguir adelante y me apoyaron en la decisión que tomé para cumplir un objetivo más.

De la misma forma, agradezco a la maestra Bety Olivares porque confió en mí, me aceptó para ser su residente en licenciatura, lo cual permitió que conociera más acerca de la maestría y quien me encaminó para continuar con mis estudios e iniciar un posgrado. Así como, a las maestras Celia Romero y María Antonieta Abud, maestro Luis Ángel Reyes, doctores Giner Alor, Ulises Juárez y a mi tutor José Luis Sánchez, por todo el apoyo que me brindaron y conocimiento.

También quiero agradecer especialmente, a la Dra. Lisbeth Rodríguez Mazahua por aceptarme como su tesista, por la paciencia y apoyo que me tuvo en todo momento, además de compartir su amplio conocimiento conmigo, tiene mis respetos y aprecio por ser una gran persona, siempre comprometida con su trabajo. Al M.S.C Felipe Castro Medina porque siempre estuvo disponible ante cualquier duda o comentario, lo cual me sirvió para culminar este proyecto.

Al Dr. Isaac Machorro Cano y Dr. Asdrúbal López Chau, gracias por permitirme realizar la estancia académica con ustedes y compartir sus recomendaciones para contribuir en el proyecto de investigación.

A mis compañeros de la M.S.C por su amistad, apoyo y buenos momentos que me hicieron pasar durante los estudios de posgrado.

Finalmente, agradezco al Consejo Nacional de Humanidades Ciencias y Tecnologías (CONAHCYT) por el apoyo económico otorgado a lo largo de estos dos años y al Tecnológico Nacional de México (TecNM) por dar soporte a esta investigación.

**Índice General**

Índice de Tablas.....	VIII
Índice de Figuras .....	X
Resumen .....	XIV
Abstract.....	XV
Introducción.....	1
Capítulo 1. Antecedentes.....	3
1.1    Marco teórico.....	3
1.1.1    Bases de datos.....	3
1.1.2    Bases de datos multimedia.....	3
1.1.3    Bases de datos distribuidas .....	3
1.1.4    Diseño de bases de datos distribuidas.....	4
1.1.5    Fragmentación en Bases de Datos .....	4
1.1.5.1    Fragmentación Horizontal (primaria y derivada).....	5
1.1.5.2    Fragmentación Vertical.....	5
1.1.5.3    Fragmentación Híbrida.....	5
1.1.6    Fragmentación Estática.....	6
1.1.7    Fragmentación Dinámica.....	6
1.1.8    Reglas de la fragmentación.....	6
1.1.9    FRAGMENT.....	7
1.1.10    XAMANA.....	7
1.2    Situación tecnológica, económica y operativa de la empresa.....	8
1.3    Planteamiento del problema.....	8
1.4    Objetivo general y específicos.....	9
1.4.1    Objetivo general.....	9
1.4.2    Objetivos específicos .....	9
1.5    Justificación .....	9
Capítulo 2. Estado de la práctica .....	11
2.1    Trabajos relacionados .....	11
2.2    Análisis comparativo.....	27

2.3	Propuesta de solución .....	43
Capítulo 3.	Aplicación de la metodología .....	49
3.1	Análisis .....	49
3.2	Selección .....	54
3.3	Desarrollo.....	55
3.3.1	Análisis de requisitos .....	60
3.3.2	Diseño .....	61
3.3.2.1	Modelo conceptual.....	62
3.3.2.2	Modelo de navegación .....	65
3.3.2.3	Modelo de presentación .....	67
3.3.2.4	Modelo de proceso .....	71
Capítulo 4.	Resultados.....	73
4.1	Resultados del análisis .....	73
4.2	Base de datos HITO.....	80
4.3	Aplicación Web XAMANA .....	82
4.3.1	Resultado de la fragmentación híbrida estática .....	91
4.4	Vigilante-fragmentador.....	95
4.4.1	Fragmentación híbrida dinámica, primer caso de estudio .....	97
4.4.2	Fragmentación híbrida dinámica, segundo caso de estudio.....	117
4.5	Comparación del rendimiento de la metodología propuesta.....	139
Capítulo 5.	Conclusiones y recomendaciones .....	154
5.1	Conclusiones .....	154
5.2	Recomendaciones .....	156
Productos Académicos	.....	157
Referencias	.....	158

## Índice de Tablas

Tabla 2.1 Análisis comparativo de los trabajos relacionados.....	27
Tabla 2.2 Alternativa de solución.....	44
Tabla 3.1 Comparación de los trabajos relacionados .....	51
Tabla 3.2 Valores asignados a tipos de operaciones .....	58
Tabla 3.3 Actores para la fragmentación híbrida dinámica.....	60
Tabla 4.1 Operaciones de la carga de trabajo para la fragmentación híbrida estática.....	86
Tabla 4.2 Ejemplo del análisis de costo de una operación para la fragmentación híbrida estática .....	89
Tabla 4.3 Operaciones de la carga de trabajo para la fragmentación híbrida dinámica, primer caso .....	97
Tabla 4.4 Análisis de costo de operaciones en <i>multimedia_records_1</i> , fragmentación híbrida dinámica, primer caso.....	102
Tabla 4.5 Comparación de los costos de los fragmentos híbridos, primer caso.....	105
Tabla 4.6 Comparación de los fragmentos verticales y horizontales en la fragmentación dinámica, primer caso de estudio .....	106
Tabla 4.7 Fragmentos formados en la fragmentación híbrida dinámica, caso 1 .....	111
Tabla 4.8 Operaciones de la carga de trabajo para la fragmentación híbrida dinámica, segundo caso .....	118
Tabla 4.9 Análisis de costo de operaciones en <i>multimedia_records_1</i> , fragmentación híbrida dinámica, segundo caso .....	122
Tabla 4.10 Comparación de los costos de los fragmentos híbridos, segundo caso .....	124
Tabla 4.11 Comparación de los fragmentos verticales y horizontales en la partición dinámica, segundo caso.....	125
Tabla 4.12 Fragmentos formados en la fragmentación híbrida dinámica, caso 2 .....	128
Tabla 4.13 Comparación del esquema de fragmentación híbrida estática con fragmentación híbrida dinámica, primer caso .....	134
Tabla 4.14 Comparación del esquema de fragmentación híbrida estática con fragmentación híbrida dinámica, segundo caso.....	135
Tabla 4.15 Comparación del esquema de fragmentación híbrida dinámica caso 1 y 2.....	137
Tabla 4.16 Operaciones de la carga de trabajo para realizar la comparación .....	141
Tabla 4.17 Resultado de la comparación con el trabajo de Castro-Medina et al. [9].....	145

Tabla 4.18 Comparación de costo de ejecución esquema vertical con híbrido .....	148
Tabla 4.19 Esquema de fragmentación híbrida dinámica, con base en el trabajo de Rodríguez-Mazahua et al. [68] .....	148
Tabla 4.20 Resultado de la comparación con el trabajo de Rodríguez-Mazahua et al. [68] ...	149

## Índice de Figuras

Figura 2.1 Estructura de la propuesta de solución.....	43
Figura 3.1 Metodología de búsqueda y evaluación de los trabajos relacionados.....	50
Figura 3.2 Flujo de trabajo de la fragmentación híbrida dinámica para bases de datos multimedia .....	56
Figura 3.3 Flujo de proceso de fragmentación híbrida dinámica para bases de datos multimedia .....	57
Figura 3.4 Arquitectura de la aplicación Web para realizar la fragmentación híbrida dinámica .....	60
Figura 3.5 Diagrama de casos de uso para la aplicación Web y la fragmentación híbrida dinámica .....	61
Figura 3.6 Diagrama de actividad del caso de uso “Fragmentar y asignar”.....	61
Figura 3.7 Diagrama conceptual de la aplicación.....	62
Figura 3.8 Diagrama lógico.....	63
Figura 3.9 Diagrama físico.....	63
Figura 3.10 Modelo de navegación de la aplicación Web.....	66
Figura 3.11 Página <i>Configuración</i> de conexión y fragmentación del modelo de presentación	68
Figura 3.12 Página <i>Estatus de conexión</i> .....	68
Figura 3.13 Página <i>Detalles de la fragmentación híbrida</i> .....	69
Figura 3.14 Página de resultados del modelo de presentación.....	70
Figura 3.15 Diagrama de proceso de la fragmentación híbrida.....	72
Figura 4.1 Número de artículos por año de publicación.....	73
Figura 4.2 Número de artículos por editorial.....	74
Figura 4.3 Número de artículos que usaron un <i>benchmark</i> .....	75
Figura 4.4 Número de artículos por tipo de <i>benchmark</i> .....	75
Figura 4.5 Número de artículos por completitud.....	76
Figura 4.6 Número de artículos por facilidad de implementación.....	76
Figura 4.7 Número de artículos por modelo de costos.....	77
Figura 4.8 Número de artículos por tipo de fragmentación dinámica o estática.....	78
Figura 4.9 Número de artículos por tipo de repositorio.....	79
Figura 4.10 Número de artículos por consultas basadas en contenido.....	79

Figura 4.11 Colecciones de la base de datos HITO, parte 1 .....	80
Figura 4.12. Colecciones de la base de datos HITO, parte 2.....	81
Figura 4.13 Un documento de la colección <i>multimedia_records</i> .....	82
Figura 4.14 Página principal de la aplicación Web XAMANA .....	83
Figura 4.15 Página de configuración.....	83
Figura 4.16 Modal de configuración exitosa .....	84
Figura 4.17 Página de configuración para la fragmentación .....	84
Figura 4.18 Modal para la asignación de tamaño de atributos .....	85
Figura 4.19 Escoger el archivo de operaciones ( <i>logs</i> ).....	86
Figura 4.20 Fragmento del archivo <i>log</i> para realizar la fragmentación híbrida estática .....	87
Figura 4.21 Asignación de umbral de operaciones y desempeño.....	87
Figura 4.22 Configuración asignada por el DBA para fragmentación híbrida estática.....	88
Figura 4.23 Mensaje con el esquema menos costoso .....	88
Figura 4.24 Resultado final de la fragmentación híbrida estática .....	89
Figura 4.25 Aplicación de la fragmentación híbrida estática con el <i>token</i> correspondiente .....	90
Figura 4.26 Resultado de la consulta tabla <i>join</i> fragmento .....	91
Figura 4.27 Fragmentos resultantes de la fragmentación híbrida estática en el sitio remoto ....	92
Figura 4.28 Colecciones de HITO en el sitio local ahora sin “ <i>multimedia_records</i> ” .....	92
Figura 4.29 Fragmento “ <i>multimedia_records_1</i> ” .....	93
Figura 4.30 Fragmento “ <i>multimedia_records_2</i> ” .....	93
Figura 4.31 Fragmento “ <i>multimedia_records_3</i> ” .....	94
Figura 4.32 Fragmento “ <i>multimedia_records_4</i> ” .....	94
Figura 4.33 Vigilantes de fragmentación supervisando fragmentos por sitio .....	95
Figura 4.34 Inicio del vigilante-fragmentador.....	96
Figura 4.35 Carga de trabajo para la fragmentación híbrida dinámica, sitio remoto .....	97
Figura 4.36 Fragmento del primer archivo <i>log</i> de “ <i>multimedia_records_1</i> ” .....	98
Figura 4.37 Carga de trabajo para la fragmentación híbrida dinámica, sitio local.....	99
Figura 4.38 Vigilante-fragmentador, sitio remoto.....	99
Figura 4.39 Vigilante-fragmentador, sitio local .....	100
Figura 4.40 Fragmentos que alcanzaron el umbral de operaciones, primer caso.....	101

Figura 4.41 Análisis de operaciones del vigilante-fragmentador, sitio local .....	101
Figura 4.42 Análisis de operaciones del vigilante-fragmentador, sitio remoto.....	102
Figura 4.43 Información del costo del fragmento “ <i>multimedia_records_1</i> ”, primer caso .....	105
Figura 4.44 Esquema vertical y horizontal dinámico considerando “ <i>multimedia_records_1</i> ”, primer caso .....	106
Figura 4.45 Esquema de la fragmentación híbrida dinámica “ <i>multimedia_records_1</i> ”, primer caso .....	108
Figura 4.46 Esquema de la fragmentación híbrida dinámica “ <i>multimedia_records_2_1</i> ” y “ <i>multimedia_records_2_1</i> ”, primer caso.....	108
Figura 4.47 Asignar el nuevo fragmento híbrido dinámico “ <i>multimedia_records_1</i> ”, primer caso .....	109
Figura 4.48 Asignar el nuevo fragmento híbrido dinámico “ <i>multimedia_records_2_1</i> ” y “ <i>multimedia_records_2_2</i> ”, primer caso.....	109
Figura 4.49 Actualización de vigilante en sitios, primer caso.....	109
Figura 4.50 Consulta sobre la base de datos XAMANA, fragmentación híbrida estática .....	110
Figura 4.51 Consulta sobre la base de datos XAMANA, fragmentación híbrida dinámica....	110
Figura 4.52 Eliminación del origen de los fragmentos primer caso, sitio 2 .....	113
Figura 4.53 Asignación de los fragmentos híbridos en el sitio local, primer caso .....	113
Figura 4.54 Fragmento “ <i>multimedia_records_1</i> ”, fragmentación híbrida dinámica, primer caso .....	114
Figura 4.55 Fragmento “ <i>multimedia_records_2_1</i> ”, fragmentación híbrida dinámica, primer caso .....	114
Figura 4.56 Fragmento “ <i>multimedia_records_2_2</i> ”, fragmentación híbrida dinámica, primer caso .....	115
Figura 4.57 Fragmento “ <i>multimedia_records_3_1</i> ”, fragmentación híbrida dinámica, primer caso .....	115
Figura 4.58 Fragmento “ <i>multimedia_records_3_2</i> ”, fragmentación híbrida dinámica, primer caso .....	116
Figura 4.59 Fragmento “ <i>multimedia_records_4_1</i> ”, fragmentación híbrida dinámica, primer caso .....	116
Figura 4.60 Fragmento “ <i>multimedia_records_4_2</i> ”, fragmentación híbrida dinámica, primer caso .....	117

Figura 4.61 Carga de trabajo para la fragmentación híbrida dinámica, sitio remoto, segundo caso .....	117
Figura 4.62 Operación de “ <i>multimedia_records_4</i> ”, segundo caso .....	118
Figura 4.63 Vigilante-fragmentador, sitio remoto, segundo caso .....	119
Figura 4.64 Fragmentos que alcanzaron el umbral de operaciones, segundo caso .....	120
Figura 4.65 Fragmento que no alcanzó el umbral de operaciones, segundo caso .....	120
Figura 4.66 Análisis de operaciones del vigilante-fragmentador, sitio remoto, segundo caso	121
Figura 4.67 Información del costo del fragmento “ <i>multimedia_records_1</i> ”, segundo caso ..	123
Figura 4.68 Esquema vertical y horizontal dinámico considerando “ <i>multimedia_records_1</i> ”, segundo caso .....	124
Figura 4.69 Esquema de la fragmentación híbrida dinámica “ <i>multimedia_records_1_1</i> ” y “ <i>multimedia_records_2_1</i> ”, segundo caso .....	126
Figura 4.70 Asignación de los nuevos fragmentos híbridos “ <i>multimedia_records_1_1</i> ” y “ <i>multimedia_records_1_2</i> ”, segundo caso .....	126
Figura 4.71 Actualización de vigilante, sitio 1, segundo caso .....	127
Figura 4.72 Sitio 2, con el fragmento “ <i>multimedia_records_4</i> ”, segundo caso .....	129
Figura 4.73 Asignación de los fragmentos híbridos en el sitio local, segundo caso .....	129
Figura 4.74 Fragmento “ <i>multimedia_records_1_1</i> ”, fragmentación híbrida dinámica, segundo caso .....	130
Figura 4.75 Fragmento “ <i>multimedia_records_1_2</i> ”, fragmentación híbrida dinámica, segundo caso .....	130
Figura 4.76 Fragmento “ <i>multimedia_records_2_1</i> ”, fragmentación híbrida dinámica, segundo caso .....	131
Figura 4.77 Fragmento “ <i>multimedia_records_2_2</i> ”, fragmentación híbrida dinámica, segundo caso .....	131
Figura 4.78 Fragmento “ <i>multimedia_records_3_1</i> ”, fragmentación híbrida dinámica, segundo caso .....	132
Figura 4.79 Fragmento “ <i>multimedia_records_2_2</i> ”, fragmentación híbrida dinámica, segundo caso .....	132
Figura 4.80 Esquema de fragmentación horizontal estática .....	140
Figura 4.81 Esquema de fragmentación vertical estática .....	141
Figura 4.82 Comparación gráfica de la fragmentación híbrida .....	153

## Resumen

Las bases de datos multimedia son de gran importancia en los sistemas de información, ya que permiten almacenar una gran cantidad de datos de diferentes tipos. Esencialmente, las bases de datos multimedia, además de incluir texto, almacenan y recuperan audio, imagen y video, por lo cual, los datos multimedia al ser de mayor tamaño, representan un reto para la recuperación eficiente de la información, ya que provocan el aumento del costo de transmisión de datos y los tiempos de respuesta.

Para resolver este problema, existen técnicas de fragmentación de datos que permiten mejorar el desempeño de las consultas, aumentar la disponibilidad de la información y ejecutar eficientemente más operaciones accediendo menos a datos irrelevantes.

El objetivo del presente trabajo fue desarrollar un método de fragmentación híbrida dinámica que adapte el esquema de acuerdo con los cambios en la carga de trabajo para mantener la recuperación eficiente de datos multimedia. Para llegar a la solución, primero se realizó un análisis comparativo de las técnicas propuestas en la literatura, de los métodos de fragmentación híbrida estáticos y dinámicos para bases de datos multimedia, así como de los modelos de costos utilizados para evaluar esquemas de fragmentación; posteriormente se diseñó e implementó la técnica propuesta y, por último, se comparó con un enfoque seleccionado del análisis comparativo. La contribución principal del presente trabajo es una técnica efectiva para la fragmentación híbrida que se integró a la aplicación Web XAMANA, la cual se validó con la base de datos HITO (Historia del Instituto Tecnológico de Orizaba).

Las tecnologías empleadas son Java como lenguaje de programación, el marco de trabajo JSF (*JavaServer Faces*), MySQL y MongoDB como sistemas gestores de bases de datos, NetBeans como IDE (*Integrated Development Environment*, entorno de desarrollo integrado) y UWE (*Unified Modeling Language Web Engineering*, Ingeniería Web del Lenguaje Unificado de Modelado) como metodología de desarrollo.

## **Abstract**

Multimedia databases are very important in information systems, because they can storage high-volume data of different types. Essentially, multimedia databases in addition to containing text, they also include images videos and audio, so, multimedia data has high size, representing a challenge to efficient information retrieval, since they cause increases in transmission data costs and response time of the queries.

To solve this problem, data fragmentation techniques exist to improve query performance, increase information availability, and efficiently execute more operations accessing less irrelevant data.

The objective of this work was to developed a dynamic hybrid fragmentation method to adapt the scheme according to workload changes to maintain efficient retrieval of multimedia data. To reach the solution, first we performed a comparative analysis of the static and dynamic hybrid fragmentation techniques for multimedia databases proposed in the literature as well as costs models used to evaluated the fragmentation scheme; subsequently we designed and implemented the proposed technique and finally, we compared it with an approach selected of the comparative analysis. The main contribution of this work is an effective technique for hybrid fragmentation that was integrated to the XAMANA Web application, which was validated using the HITO database (History of the Technological Institute of Orizaba).

The technologies employed was Java as a programming language, JavaServer Faces (JSF) as a framework, MySQL and MongoDB database management systems, and NetBeans as an Integrated Development Environment (IDE), following the UWE methodology (Unified Modeling Language-based Web Engineering).

## Introducción

En la actualidad, el crecimiento de los datos multimedia dificulta la administración de las bases de datos al momento de realizar consultas, provocando el aumento del tiempo de respuesta y el costo de ejecución de las operaciones. La fragmentación de datos es una técnica de diseño en bases de datos multimedia que permite reducir ambas cifras.

Existen tres tipos de fragmentación: la fragmentación horizontal divide una relación a lo largo de tuplas (filas), de tal manera que cada fragmento tenga un subconjunto de las tuplas de la relación; la fragmentación vertical consiste en dividir una relación en subconjuntos de atributos (columnas) y en cada fragmento colocar la llave primaria de la relación para permitir su reconstrucción; y, por último, la fragmentación híbrida que es la combinación de la horizontal-vertical o viceversa. Además, estos métodos se clasifican en estáticos y dinámicos, son pocos los trabajos que abordan una fragmentación dinámica, es decir, modificar el esquema continuamente con base en las consultas realizadas para mantener la recuperación eficiente de datos multimedia.

En esta investigación se abordó el problema proponiendo un método para la fragmentación híbrida dinámica. Por consiguiente, el método desarrollado es capaz de aplicar el esquema menos costoso (horizontal-vertical o vertical-horizontal) y de adaptar el esquema a nuevas cargas de trabajo, obteniendo como beneficio mejores resultados en la recuperación de los datos multimedia, además, de contribuir al estado del arte con un nuevo enfoque para incrementar los trabajos relacionados con la fragmentación híbrida dinámica.

El presente trabajo se organiza en cinco capítulos, el capítulo uno aborda los antecedentes del proyecto que incluye el marco teórico, planteamiento del problema, situación tecnológica, económica y operativa de la empresa, objetivo general y específicos y la justificación; en el capítulo dos se presenta el estado de la práctica, análisis comparativo de los trabajos relacionados con el tema y la propuesta de solución considerando el marco de trabajo, IDE, sistemas gestores de bases de datos y la metodología Web para el desarrollo del método. El tercer capítulo presenta un análisis profundo de los métodos de fragmentación híbrida dinámica, el diseño de la aplicación Web y el flujo del vigilante-fragmentador considerando los modelos

de costos, siguiendo la metodología de UWE. El cuarto capítulo describe los resultados obtenidos por la aplicación Web XAMANA y el vigilante-fragmentador, además de incluir dos casos de estudio para validar su funcionamiento. Finalmente, en el capítulo cinco se presentan las conclusiones y recomendaciones.

## **Capítulo 1. Antecedentes**

En este primer capítulo se presentan algunos términos relevantes para el trabajo de investigación; el problema a resolver, el objetivo general y los objetivos específicos, así como la justificación de lo que se realizará.

### **1.1 Marco teórico**

A continuación, se definen algunos términos que se utilizarán para el trabajo de investigación.

#### **1.1.1 Bases de datos**

Las bases de datos son una colección autodescriptiva de registros integrados. Un registro es una representación de algún objeto físico o conceptual [1]. Por ejemplo, el conjunto de datos que incluye la matrícula, el nombre, la carrera, el semestre, la edad y el sexo de un estudiante forman el registro de ese estudiante. El propósito de una base de datos es recopilar, almacenar y recuperar información.

De acuerdo al tipo de información que se almacene, será la base de datos a emplear; en este trabajo de investigación se utilizarán bases de datos multimedia.

#### **1.1.2 Bases de datos multimedia**

Las bases de datos multimedia son una colección de datos, que además de incluir texto, almacenan y recuperan audio, imagen y video. Al incluir datos cuya principal característica es su gran tamaño, resulta complicada la gestión óptima de los mismos por el crecimiento del contenido multimedia; por consiguiente, es necesario aplicar alguna técnica de fragmentación, de tal manera que se conserve la calidad de los datos almacenados, pero la gestión sea óptima [2].

#### **1.1.3 Bases de datos distribuidas**

Una base de datos distribuida es una colección de múltiples bases de datos interrelacionadas que se encuentran situadas en espacios físicos o lógicos, por una red informática [3]. La distribución física no implica necesariamente que los sistemas estén separados geográficamente; en

ocasiones se encuentran en la misma habitación. Simplemente implica que la comunicación entre ellos se realice a través de una red en lugar de memoria compartida o disco compartido.

Las bases de datos distribuidas se caracterizan por almacenar la información en varias computadoras conectadas entre sí, a las cuales el usuario accede desde cualquier sitio como si se tratara de una red local.

#### **1.1.4 Diseño de bases de datos distribuidas**

En el diseño de bases de datos distribuidas se considera cómo se va a distribuir la información entre diferentes sitios. Sin embargo, cuando se busca eficiencia en el acceso de la información, es importante analizar: cómo fragmentar la información y la manera de asignar cada fragmento entre los sitios en la red; así mismo, tomar en cuenta si la información está replicada, es decir, si existen copias múltiples del mismo dato y, en este caso, cómo mantener la consistencia de la información. Finalmente, una parte importante de las bases de datos se refiere al manejo del directorio. Si existen únicamente usuarios globales, se maneja un solo directorio global. Sin embargo, si existen también usuarios locales, el directorio combina información local con información global [3].

#### **1.1.5 Fragmentación en Bases de Datos**

La fragmentación de bases de datos es un proceso para reducir los accesos a datos irrelevantes mediante la agrupación de datos a los que se accede con frecuencia en segmentos dedicados [4]; de tal manera, se reducen los tiempos de respuesta, provocando menores costos de ejecución en cada operación realizada.

Se definieron tres técnicas principales de fragmentación para las bases de datos relacionales:

1. Fragmentación horizontal (HF).
2. Fragmentación vertical (VF).
3. Fragmentación híbrida o mixta (MF).

### **1.1.5.1 Fragmentación Horizontal (primaria y derivada)**

La fragmentación horizontal divide una relación a lo largo de las tuplas. Por ello cada fragmento es un subconjunto de tuplas de esa relación. Hay dos versiones de la fragmentación horizontal: la primaria y la derivada [3].

1. **Fragmentación horizontal primaria:** se realiza usando predicados que definen una relación.
2. **Fragmentación horizontal derivada:** es la fragmentación de una relación como resultado de predicados que se definen en otra relación.

La fragmentación horizontal utiliza el operador de selección donde los predicados determinan la fragmentación.

### **1.1.5.2 Fragmentación Vertical**

La fragmentación vertical consiste en dividir una relación en subconjuntos de atributos y en cada fragmento colocar la llave primaria de la relación para permitir su reconstrucción. El objetivo de la fragmentación vertical es dividir una relación en un conjunto de pequeñas relaciones para que la mayor parte de las aplicaciones se ejecuten en un solo fragmento. La fragmentación vertical se investiga en el contexto de sistemas de bases de datos centralizadas y distribuidas, y se desempeña como una herramienta de diseño que permite a las consultas de los usuarios tratar con relaciones más pequeñas [3].

La fragmentación vertical se realiza por medio del operador de proyección.

### **1.1.5.3 Fragmentación Híbrida**

En muchos casos una fragmentación horizontal o vertical de un esquema de base de datos no es suficiente para satisfacer los requerimientos de las aplicaciones. En estos casos una fragmentación vertical se lleva a cabo después de una horizontal o viceversa, produciendo un árbol estructurado de particiones, a esta técnica se le llama fragmentación híbrida, fragmentación mixta o fragmentación anidada [3]. El enfoque de fragmentación híbrida utiliza la fragmentación horizontal y vertical [5]; a continuación, se describen estos dos métodos:

**Híbrido Horizontal-Vertical:** primero genera un conjunto de fragmentos horizontales; luego se generan fragmentos verticales a partir de uno o más de los fragmentos horizontales.

**Híbrido Vertical-Horizontal:** primero genera un conjunto de fragmentos verticales; luego se generan fragmentos horizontales a partir de uno o más de los fragmentos verticales.

### 1.1.6 Fragmentación Estática

En la fragmentación estática, los datos o elementos de una tabla (atributos y/o tuplas) se asignan a un fragmento solo una vez en el momento de la creación, luego sus ubicaciones nunca cambian [6].

### 1.1.7 Fragmentación Dinámica

La fragmentación dinámica mejora el rendimiento de las bases de datos, resolviendo diferentes problemas de un enfoque estático. En la fragmentación dinámica, los elementos de una relación (atributos y/o tuplas) se reubican si se detecta que el esquema actual de fragmentación se ha vuelto inadecuado por los cambios en la información de las consultas [6].

### 1.1.8 Reglas de la fragmentación

Hay que tener en cuenta que la fragmentación afecta al rendimiento del SGBD (Sistema Gestor de Bases de Datos), especialmente cuando se utilizan distintos fragmentos ubicados en diferentes nodos para construir una vista.

A continuación, se presentan las tres reglas de fragmentación que, juntas, garantizan que las bases de datos no experimenten cambios semánticos durante la fragmentación [3].

- I. **Compleitud:** Si una instancia de relación  $R$  se descompone en fragmentos  $F_R = \{R_1, R_2, \dots, R_n\}$ , cada dato que está en  $R$  también se encuentra en uno o más de  $R_i$ 's. Esta propiedad, que es idéntica a la propiedad de descomposición sin pérdidas de la normalización, también es importante en la fragmentación, ya que asegura que los datos en una relación global se mapean en fragmentos sin ninguna pérdida [7]. Hay que considerar que, en el caso de la fragmentación horizontal, el "elemento" generalmente

se refiere a una tupla, mientras que, en el caso de la fragmentación vertical, se refiere a un atributo.

- II. Reconstrucción:** Si una relación  $R$  se descompone en fragmentos  $F_R = \{R_1, R_2, \dots, R_n\}$ , es posible definir un operador relacional  $\nabla$  tal que,

$$R = \nabla R_i, \quad \forall R_i \in F_R$$

El operador  $\nabla$  será distinto para diferentes formas de fragmentación, es decir, para la fragmentación horizontal este operador es la unión y en la vertical la reunión. La reconstrucción de la relación a partir de sus fragmentos garantiza que se conserven las restricciones definidas sobre los datos en forma de dependencias.

- III. Disyunción:** Si una relación  $R$  se descompone horizontalmente en fragmentos  $F_R = \{R_1, R_2, \dots, R_n\}$  y el dato  $d_i$  está en  $R_j$ , entonces no está en ningún otro fragmento  $R_k$  ( $k \neq j$ ). Este criterio asegura que los fragmentos horizontales sean disjuntos. Si la relación  $R$  se descompone verticalmente, sus atributos de clave primaria se repiten normalmente en todos sus fragmentos (para la reconstrucción). Por lo tanto, en el caso de la fragmentación vertical, la disyunción se define solo en los atributos que no formen parte de la clave primaria de una relación.

### 1.1.9 FRAGMENT

FRAGMENT es una aplicación Web desarrollada bajo el lenguaje de programación Java con el marco de trabajo *JavaServer Faces*, que permite aplicar un esquema adecuado de fragmentación horizontal sobre sitios en la nube, realizando las operaciones de asignación y replicación automáticamente, usando como sistema gestor de bases de datos MySQL [8].

### 1.1.10 XAMANA

XAMANA es una aplicación Web desarrollada bajo el lenguaje de programación Java con JSF, encargada de aplicar los tres tipos de fragmentación: fragmentación horizontal, vertical e híbrida, contemplando CBIR (*Content-Based Image Retrieval*, Consulta de Imágenes Basada en Contenido). Los SGBD que contempla son: MySQL, PostgreSQL, Postgres-XL y MongoDB [9].

## **1.2 Situación tecnológica, económica y operativa de la empresa**

El Instituto Tecnológico de Orizaba es una institución pública, que pertenece al Tecnológico Nacional de México. Se encuentra ubicado en Avenida Oriente 9 Núm. 852, Colonia Emiliano Zapata, C.P. 94320, Orizaba, Veracruz. Brinda educación superior, orientada al desarrollo de la ciencia y la técnica. Actualmente, en el Instituto se imparten carreras a nivel licenciatura y estudios de posgrado. A nivel licenciatura, cuenta con ocho programas educativos.

La División de Estudios de Posgrado e Investigación del Instituto Tecnológico de Orizaba cuenta con cinco maestrías y un Doctorado en Ciencias de la Ingeniería, dentro de las cinco maestrías se encuentra la Maestría en Sistemas Computacionales.

## **1.3 Planteamiento del problema**

Las bases de datos multimedia además de contener texto, almacenan contenido multimedia teniendo como principal característica su gran tamaño; por lo cual, al momento de realizar consultas, los tiempos de respuesta de las consultas y ejecución son tardados. La fragmentación es una técnica de diseño muy utilizada en bases de datos multimedia, que reduce los tiempos de respuesta de las consultas y como resultado tienen menores costos de ejecución. La mayoría de los métodos existentes en la literatura se enfocan en la fragmentación horizontal o vertical. Generalmente, los métodos que consideran la fragmentación híbrida no son capaces de adaptar el esquema de fragmentación a los patrones de acceso a las bases de datos multimedia.

Por tal motivo, en este proyecto se desarrolló un método de fragmentación híbrida dinámica para bases de multimedia que es capaz de obtener un nuevo esquema cuando ocurren suficientes cambios en la carga de trabajo, para así garantizar el buen desempeño de las consultas en todo momento. Para esto, se realizó una búsqueda de los artículos relacionados con este tema en las principales bibliotecas digitales, como ACM digital library, SpringerLink, ScienceDirect y IEEE Xplore; revisando los métodos de fragmentación híbrida dinámica aplicados en bases de datos multimedia para determinar si existe uno que considere la reducción de costos y tiempos de ejecución de las consultas y además sea fácil de implementar y contenga todo lo necesario para su replicación. La técnica de fragmentación híbrida se diseñó e implementó. Finalmente,

el enfoque seleccionado en el análisis comparativo se utilizó para evaluar el método propuesto en este proyecto.

## **1.4 Objetivo general y específicos**

A continuación, se presenta el objetivo general y los objetivos específicos.

### **1.4.1 Objetivo general**

Desarrollar un método de fragmentación híbrida dinámica que adapte el esquema de acuerdo con los cambios en la carga de trabajo para que se mantenga la recuperación eficiente de datos multimedia.

### **1.4.2 Objetivos específicos**

1. Analizar el estado del arte de los métodos de fragmentación híbrida estáticos y dinámicos, para bases de datos multimedia, así como de los modelos de costos utilizados para la evaluación de esquemas de fragmentación.
2. Realizar un análisis comparativo de los algoritmos y de los modelos de costos identificados en el estado del arte para que se determinen sus ventajas y desventajas, así como las bases de datos en las que se implementaron.
3. Seleccionar las tecnologías que se utilizarán para el desarrollo del método.
4. Diseñar el método y el modelo de costos para la fragmentación híbrida dinámica de bases de datos multimedia.
5. Implementar el método y modelo de costos utilizando las tecnologías seleccionadas para que se logre la reducción constante del tiempo y costo de ejecución de consultas.
6. Comparar el método y modelo de costos con un algoritmo y modelo seleccionados del análisis comparativo usando un caso de estudio para su validación.

## **1.5 Justificación**

Debido al crecimiento exponencial de datos multimedia en diversas aplicaciones, los métodos de fragmentación híbrida cada vez son más frecuentes en bases de datos multimedia [5], [10]. Estos métodos se clasifican en estáticos y dinámicos, los primeros obtienen un esquema de fragmentación de acuerdo con una carga de trabajo, pero si esta cambia, el esquema no es capaz

de adaptarse a las nuevas consultas para seguir siendo adecuado. Los métodos dinámicos resuelven este problema, ya que modifican el esquema continuamente con base en las consultas [2].

Por medio de este trabajo, al desarrollar un método de fragmentación híbrida que logre adaptar el esquema de acuerdo con los cambios en la carga de trabajo, se obtendrán mejores resultados en la recuperación de los datos multimedia. Además, se logrará contribuir al estado del arte de los métodos de fragmentación para que los investigadores del área de bases de datos cuenten con una nueva técnica que utilicen para compararla con sus propuestas.

Finalmente, este proyecto integrará los métodos de fragmentación horizontal y vertical desarrollados en tesis anteriores ([11], [12]).

## Capítulo 2. Estado de la práctica

Antes de introducirse al tema por completo, se revisó un total de 15 artículos de forma detallada, de los cuales se tomó la información más relevante, considerando las investigaciones relacionadas con el tema propuesto. El análisis de los artículos dio como resultado la demostración de la importancia que tiene la aplicación de un método de fragmentación híbrida dinámica en bases de datos multimedia; de tal manera que, cada artículo consultado contribuyó a la presente investigación.

A continuación, se presentan los aspectos más importantes de cada artículo estudiado.

### 2.1 Trabajos relacionados

Para mejorar el desempeño y el costo de las operaciones en un ambiente distribuido de bases de datos relacionales se usan las técnicas de fragmentación, asignación y replicación. Castro-Medina et al. [8] presentaron un método de fragmentación y replicación que se aplicó a un entorno en la nube por medio de FRAGMENT. Esta es una aplicación Web que tiene por objetivo aplicar un esquema adecuado de fragmentación sobre sitios en la nube, realizando las operaciones de asignación y replicación automáticamente, que se enfocó en mejorar el rendimiento de las operaciones ejecutadas sobre la base de datos. Los esquemas de fragmentación se producen mediante la técnica propuesta en Abdel et al. [13], para obtener la información requerida por ésta, se analiza el archivo de carga de la base de datos. En la primera etapa se obtuvieron los datos de la conexión y el archivo de carga de la base de datos centralizada. Después de analizar los datos, mediante el archivo de carga se crea la MCRUD (*Modified Create, Read, Update and Delete*, Matriz modificada de creación, lectura, actualización y eliminación). En la siguiente etapa, por medio de la MCRUD se determinó la tabla ALP (*Attribute Locality Precedence*, Precedencia de Localidad de Atributo). Junto a la tabla ALP también se muestra el esquema propuesto de fragmentación, asignación y replicación, siendo éste la siguiente etapa del flujo de trabajo. Si se presentan dos máximos valores ALP, es posible cambiar el atributo principal y de esta manera modificar el esquema de fragmentación, asignación y replicación propuesto. Para la validación de la aplicación Web se presentaron dos casos de estudio, uno de ellos basado en una aplicación Web alterna dedicada a la simulación

de operaciones y sitios, y el otro basado en un ambiente real en la nube sobre AWS (*Amazon Web Services*, Servicios Web de Amazon). El esquema propuesto por FRAGMENT resuelve el problema de los fragmentos traslapados, priorizando predicados de mayor tamaño lógico. Se creó el simulador de operaciones con CloudSim para validar la implementación sobre el *benchmark* TPC-E y, por último, se utilizó un ambiente real sobre *Amazon Web Services*. Al solucionar el problema de los fragmentos traslapados se obtuvo un esquema adecuado con fragmentos disjuntos y completos, a partir de la relación original.

La fragmentación en las bases de datos se logra a través de un diseño adecuado en ambientes distribuidos como la nube, que beneficia a las operaciones de lectura y escritura. En Castro Medina et al. [14] se abordó de otra forma el problema del traslape de fragmentos en la aplicación FRAGMENT [8]. En el flujo de la aplicación al generar la tabla ALP se pierde la especificación de los costos de cada predicado, ya que los costos se calculan por atributo y, en general, obtener los fragmentos utilizando el atributo más costoso no determina la precedencia de los predicados. Los autores mostraron una mejora a FRAGMENT resolviendo el problema del traslape de fragmentos mediante la precedencia de costos obtenida de la MCRUD. Anteriormente se propuso eliminar los predicados más pequeños contenidos en predicados más grandes. El problema del traslape se resolvió mejorando el algoritmo de fragmentación. La mejora consiste en ordenar los predicados por costo y de esta manera obtener cada predicado contemplando que produzca fragmentos traslapados, sin embargo, los fragmentos con mayor costo se mantendrán más intactos que los de menor costo. Se mejoró el esquema producido al reducir el tiempo de ejecución de las consultas relacionadas con el atributo más costoso en comparación con el esquema producido con el enfoque anterior. Finalmente, la preservación de fragmentos se basó en la prioridad del costo y no en el tamaño, ya que en muchos casos el fragmento más grande es el menos utilizado o el fragmento más pequeño es el más utilizado.

El almacenamiento y manejo adecuado de datos multimedia es un tema de gran interés para la industria y la academia, ya que se recopilan grandes cantidades de datos complejos (p. ej., imagen, sonido y video); estos datos complejos vienen en diferentes formatos y no se ordenan por contenido de la misma manera que los datos convencionales (aquellos representados como cadenas o números). Los patrones de acceso a las bases de datos multimedia cambian

constantemente; por lo tanto, es importante que los esquemas de partición también se adapten a estos cambios. Rodríguez-Mazahua et al. [6] mostraron diferentes desventajas del enfoque estático y mencionan el enfoque dinámico como una solución para todos ellos, por lo tanto, en Castro-Medina et al. [9] se propuso un nuevo método de fragmentación horizontal dinámica para bases de datos multimedia que contempla consultas basadas en contenido y determina cuándo realizar nuevas fragmentaciones en bases de datos distribuidas. El método incluido en [9] se ubicó dentro de una aplicación Web llamada XAMANA. En el primer paso se obtuvo el nombre del SGBD que utiliza el usuario y los datos para establecer una conexión remota con privilegios de lectura y escritura. Después de verificar si las consultas basadas en contenido se ejecutaron en la tabla que se va a fragmentar, se agrupan los descriptores alojados en la columna que elige el usuario; en esta etapa se solicitaron los datos que forman parte del flujo de trabajo de la fragmentación horizontal dinámica. En la penúltima etapa se presentó el esquema propuesto, si el usuario está de acuerdo con este, pasará a la última etapa. La fragmentación dinámica se lleva a cabo en la biblioteca denominada observador-fragmentador; que utiliza la gráfica de la curva gaussiana para normalizar el uso de las tuplas a lo largo del fragmento por los diferentes sitios. La consideración de las bases de datos multimedia y CBIR utilizando consultas k-NN (*K-Nearest Neighbor*, K-Vecino más Cercano) y descriptores SURF (*Speeded Up Robust Features*, Características Robustas Aceleradas) hace este trabajo un estudio valioso para enfrentar desafíos y utilizar la fragmentación como un enfoque para mejorar el rendimiento de este tipo de bases de datos.

Las imágenes son uno de los tipos de datos más populares y atractivos que los usuarios generan o necesitan. Las imágenes médicas son los datos más importantes utilizados para diferentes propósitos (por ejemplo, diagnósticos, tratamientos, investigaciones, por mencionar algunos) en el sector de la salud. En Asghar-Safaei [5] se propuso una técnica de indexación de imágenes médicas basada en texto que utiliza los atributos de las imágenes médicas y los fragmenta, de manera híbrida (horizontal-vertical), permitiendo reformar cada uno de esos fragmentos de atributos en una jerarquía y construyendo un índice multidimensional. El proceso de indexación es una de las partes más importantes e influyentes de la recuperación de información, la cual consiste en registrar ordenadamente datos para elaborar su índice. Para indexar imágenes existen

dos enfoques principales: indexación basada en contenido y basada en texto. En [5] se utilizó la técnica de indexación basada en texto que, en lugar de crear un índice único o un conjunto de múltiples índices distintos, se construye y utiliza un índice multidimensional. La indexación basada en texto se aplicó mediante atributos anotados de las imágenes, p. ej., como atributos en el formato estándar DICOM (*Digital imaging and communications in medicine*, Imágenes digitales y comunicación en medicina) o mediante la extracción de características de la imagen. El rendimiento de la técnica de indexación multidimensional basada en fragmentación híbrida propuesta se evaluó experimentalmente al implementarla en el motor de búsqueda de código abierto Lucene [15], considerando las métricas más importantes de efectividad (precisión y recuperación) además de la eficiencia (tiempo de respuesta). Los resultados de la evaluación mostraron que, en términos de eficiencia, la técnica de indexación multidimensional basada en fragmentación híbrida superó a la técnica de indexación de Lucene.

La partición de datos es una característica esencial para ajustar las bases de datos relacionales y sigue siendo un problema abierto para los sistemas NewSQL. Schreiner et al. [16] propusieron un enfoque de fragmentación híbrida para bases de datos NewSQL que permite al usuario definir las particiones de datos verticales y horizontales. NewSQL es una nueva generación de bases de datos que proporciona alta escalabilidad, disponibilidad y compatibilidad con propiedades ACID (*Atomicity, Consistency, Isolation and Durability*, Atomicidad, Consistencia, Aislamiento y Durabilidad). Dado que las bases de datos NewSQL se distribuyen en memoria, manejan miles de transacciones OLTP (*Online Transaction Processing*, Procesamiento de Transacciones en Línea), por lo tanto, estas mantienen particiones de los datos en varios nodos [17]–[21], que todavía sufren cuando se requieren transacciones multisitio (distribuidas), ya que las transacciones generalmente se serializan y se ejecutan, cuando es posible, en un solo sitio; cuando una transacción necesita acceder a datos en varios sitios, el rendimiento del sistema se degrada. La partición es una técnica que se utiliza para mitigar la degradación del rendimiento de la transacción en varios sitios. Cada partición almacena datos relacionados entre sí, provocando que las transacciones se ejecuten localmente. El uso de particiones híbridas en aplicaciones OLTP sigue siendo un problema abierto para las bases de datos NewSQL. Hybrid VoltDB, es una nueva estrategia de partición híbrida para sistemas NewSQL desarrollada sobre

la arquitectura VoltDB. Se aprovechó la función *hash* de rango original, generalmente implementada para la partición horizontal, para implementar nuevos enfoques de partición vertical y, en consecuencia, híbrida. Para proporcionar la fragmentación híbrida: como primer punto se extendió su DDL (*Data Definition Language*, Lenguaje de definición de datos) para permitir la definición de partición vertical; como segundo se modificó la función *hash* para considerar la información del esquema y las estadísticas de acceso para dividir una tabla en fragmentos verticales almacenándose en diferentes sitios; y como tercero, cambió la ejecución de la instrucción *SELECT* para permitir el procesamiento de los fragmentos de la tabla vertical y, cuando sea necesario, unir estos fragmentos. La contribución que se obtuvo con Hybrid-VoltDB es mejor para las bases de datos NewSQL basadas en OLTP, que solo ofrecen partición de datos horizontal. Al incluir soporte adicional a la partición vertical, se mejoró el rendimiento de las transacciones OLTP distribuidas. Todas las mejoras se evaluaron y validaron a través de un conjunto de experimentos. Los resultados mostraron la eficiencia de Hybrid-VoltDB en dos aspectos: en términos de rendimiento de transacciones distribuidas y en el procesamiento general de transacciones de VoltDB nativo, incluso sin partición de datos.

Uno de los principales problemas del diseño de bases de datos distribuidas se basa en la fragmentación de las relaciones. Dahal & Joshi [22] estudiaron el algoritmo de partición vertical basado en MST (*Minimum Spanning Tree*, Árbol de Expansión Mínima) y el algoritmo de partición vertical basado en *k*-Means (KMC, *k*-Means Clustering), con el objetivo de encontrar el mejor algoritmo de fragmentación vertical. Para realizar la comparación de los algoritmos, el principal dato requerido fue una lista de consultas del usuario. Estas consultas del usuario mostraron el comportamiento de un usuario que accede al atributo desde un sitio diferente. Se obtuvieron alrededor de 100 consultas de usuarios en función del comportamiento aleatorio de los usuarios. Se consideró una tabla con 10 atributos que se fragmenta en varios clústeres. La lista de consultas de usuarios se dividió en 10 grupos, donde cada grupo consta además de 10 consultas cada uno. Para cada grupo se obtuvieron fragmentos verticales de la relación para ambos casos. Además, se evaluó el costo total de acceso que se produjo para los fragmentos de una relación. También se utilizó la herramienta *Partition Comparator* que permite evaluar el costo total de acceso que se produjo para los fragmentos de una relación. Los resultados

experimentales mostraron que la fragmentación basada en MST tiene un valor de partición más alto que la fragmentación basada en KMC en la mayoría de los casos. Por lo tanto, la partición generada por el algoritmo basado en KMC es mucho mejor que el algoritmo basado en MST. Finalmente, el algoritmo de fragmentación vertical basado en KMC se utilizó para el diseño eficiente de una base de datos distribuida.

Hoy en día, se generan grandes cantidades de datos distribuidos en todo el mundo a través de la red de datos. La replicación es uno de los mecanismos para administrar datos, ya que mejora la accesibilidad y confiabilidad de los datos en un entorno de bases de datos distribuidas. En Noraziah et al. [23] se propuso el algoritmo *Binary Vote Assignment on Grid Quorum with Association Rule* (BVAGQ-AR) para manejar la replicación síncrona de bases de datos fragmentadas. La idea principal de la replicación es la creación de varias copias de los mismos datos o réplicas en varios recursos de almacenamiento y, aunque se centran en la replicación, existen algunos métodos que descuidan la correlación entre diferentes archivos de datos. Hay cuatro fases involucradas en el marco BAVGQ-AR, que son:

1. **Minería de datos:** se utiliza el algoritmo Apriori de Reglas de Asociación para descubrir la correlación entre los datos, a través de reglas de asociación.
2. **Fragmentación de la base de datos:** los datos se fragmentan en función de los resultados del análisis de minería de datos.
3. **Asignación de base de datos:** todos los sitios están organizados lógicamente en forma de estructura de cuadrícula bidimensional.
4. **Replicación de base de datos:** después del proceso de asignación de base de datos, cada sitio tiene un archivo de relación de base de datos.

Para mostrar el funcionamiento de BVAGQ-AR se consideraron 9 servidores que están organizados lógicamente en  $3 \times 3$ , según el diseño lógico bidimensional BVAGQ-AR. Se utilizaron 9 servidores para obtener los datos replicados ( $d$ ) máximos considerando el valor  $d = 5$ . Cada servidor o nodo está conectado entre sí a través de un concentrador de conmutador de Ethernet rápido. Para preservar la consistencia de los datos y la confiabilidad de los sistemas, la gestión de transacciones es muy importante. BVAGQ-AR resuelve esto configurando el bloqueo

con un tamaño de quórum (número mínimo de votos que una transacción distribuida tiene que obtener para realizar una operación en un sistema distribuido) pequeño antes de la actualización y confirmando la transacción de forma sincronizada con los sitios que tienen los mismos datos fragmentados. La gestión de la replicación y la transacción a través del BVAGQ-AR propuesto es capaz de preservar la coherencia de los datos. También aumentaron los grados de paralelismo porque al usar la fragmentación y la replicación, la transacción se divide en varias subconsultas que operan en los fragmentos.

Al momento de aplicar la fragmentación también se consigue proteger la privacidad de las bases de datos distribuidas al dividir los atributos de las asociaciones confidenciales en diferentes fragmentos, sin embargo, en la fragmentación dinámica se tiene que considerar la preservación de la privacidad, así como el costo de comunicación durante la optimización. Por lo tanto, en [24] se propuso solucionar este problema, desarrollando un algoritmo de evolución diferencial distribuida basado en la transferencia de conocimiento (KT-DDE) para lograr el costo de comunicación óptimo y mantener la preservación de la privacidad. El algoritmo KT-DDE desarrollado incluye un marco distribuido y un optimizador basado en DE (*Differential Evolution*, Diferencial Evolutivo). El marco distribuido se propuso para transferir conocimientos de fragmentación entre diferentes subproblemas de fragmentación de bases de datos. Los operadores de cruce y mutación en el optimizador ayudaron a lograr el equilibrio entre la búsqueda exploratoria y de explotación. También se propuso un operador de selección bidimensional para comparar individuos en términos de grado de privacidad y costo de comunicación. Para el caso del algoritmo de referencia DE, se implementaron dos variantes:

1. **DE-KT-Optimizer:** esta variante adopta los operadores de mutación y cruce basados en transferencia de conocimiento en KT-DDE.
2. **DE-KT-Framework:** en esta variante, se agrega el marco basado en la transferencia de conocimiento mientras que los operadores originales de mutación y cruce están reservados en DE.

Se investigó el desempeño de KT-DDE en problemas de fragmentación dinámica de bases de datos y se verificó la efectividad de los operadores propuestos. Como trabajo a futuro, por la

efectividad de KT-DDE, se propuso adaptar las estrategias en KT-DDE a otros problemas de fragmentación de bases de datos.

Las empresas trasladan cada vez más sus bases de datos a la nube; a raíz de esto, los proveedores de bases de datos como servicio enfrentan el desafío de cumplir con las garantías de rendimiento aseguradas en SLAs (*service-level agreements*, acuerdos de nivel de servicio), manteniendo un costo lo más bajo posible, sin embargo, la cantidad provisionada de DRAM (*Dynamic Random-Access Memory*, Memoria Dinámica de Acceso Aleatorio) domina los costos de hardware. Una forma de reducir la huella de memoria es aprovechar el sesgo de acceso en la carga de trabajo, moviendo los datos inactivos a los que rara vez se accede a capas de almacenamiento más baratas y reteniendo solo los datos activos a los que se accede con frecuencia en la memoria principal. Por lo tanto, en [25] se presentó SAHARA, un asesor que propone una partición de tablas para almacenes de columnas con un consumo mínimo de memoria y, al mismo tiempo, cumple con todos los SLAs de rendimiento; también se integró SAHARA en una base de datos comercial en la nube. SAHARA recopila accesos a datos físicos y, por lo tanto, no es sensible al sesgo en la distribución de los accesos a datos, así mismo, ayuda a la partición de tablas y maneja todos los operadores de la carga de trabajo. Las siguientes contribuciones se realizaron en SAHARA:

1. Se formalizó el problema de minimizar la huella de memoria mientras se cumplió con los SLAs de rendimiento de particiones.
2. Se integró SAHARA de manera prototípica en SAP HANA Cloud (infraestructura de base de datos como servicio DBaaS) y se analizó la huella de memoria, los costos de hardware, la precisión de las estimaciones, la optimización, los gastos generales y el tiempo de optimización para puntos de referencia sintéticos y del mundo real.
3. Se propuso un nuevo modelo de costos, calculando la huella de memoria para proponer una partición de tabla recursivamente; se calculó el tamaño de búfer para cumplir con un desempeño dado, es decir, el tiempo máximo de ejecución de la carga de trabajo; logrando reducir sustancialmente el tamaño del grupo de búfer para mantener solo las páginas con una alta densidad de datos en DRAM.

Finalmente, SAHARA es práctico para sistemas comerciales populares, libres y maduros, principalmente porque es un prototipo de una base de datos comercial en la nube y reduce la huella de memoria.

En sistemas de bases de datos distribuidas, las bases de datos se encuentran en el mismo sistema, pero geográficamente están dispersas en múltiples ubicaciones [26], permitiendo a los usuarios acceder a los datos a través de sitios locales y remotos. El acceso remoto a sitios produce un aumento en los tiempos de respuesta de las operaciones, provocando un rendimiento deficiente del sistema. Para solucionar este problema se requiere de la fragmentación vertical, por lo que en Aggarwal-Mukta et al. [27] se presentó la técnica de agrupamiento  $k$ -Medoids que se utiliza para la fragmentación vertical sin superposición. Primero se generó la AUM (*Attribute Usage Matrix*, Matriz de Uso de Atributos). Para su creación se registró la ejecución de las consultas en cada sitio y el atributo utilizado por cada consulta se tabula en AUM. Como segundo paso se aplicó el algoritmo  $k$ -Medoids en AUM. Aquí,  $k$ -Medoids se aplica en una base de datos para fragmentarla verticalmente. Para trabajar con el algoritmo, los  $K$  atributos (donde  $K$  es el número de fragmentos a formar) se eligieron aleatoriamente como *medoides*. El algoritmo minimizó la suma de las diferencias entre los atributos que se van a fragmentar y un atributo se designó como el *medoide* de un clúster. En comparación con EMST (*Enhanced Minimum Spanning Tree*, Árbol de expansión mínimo mejorado), no hay necesidad de construir la matriz ECRUD (*Enhanced Create, Update, Read and Delete*, Creación, actualización, lectura y eliminación mejoradas) y ASM (*attribute similarity matrix*, matriz de similitud de atributos) y solo se crea y utiliza la AUM. Posteriormente, se calculó un Recuento de Disimilitud (DC) para mostrar que los fragmentos formados son apropiados y que los atributos incluidos en los fragmentos son los más similares. La medida de similitud consideró cuántas veces se accede a los atributos juntos en el mismo sitio. Los experimentos mostraron que la metodología propuesta obtuvo los mismos o mejores resultados que EMST y la idoneidad de los fragmentos formados también se comprobó mediante el cálculo de DC. En comparación con EMST, el algoritmo propuesto es simple porque solo requiere una matriz AUM, usando el valor DC para la fragmentación para que  $k$ -Medoids agrupe los atributos con la menor disimilitud. Finalmente,  $k$ -Medoids, un método de agrupamiento, se utilizó para generar fragmentos no superpuestos. Los fragmentos generados

se comprueban calculando el valor de DC y los atributos se colocan sobre la base de la menor disimilitud. También es capaz de agregar nuevos atributos en fragmentos existentes comparando su diferencia con los *medoides*. Elegir *medoides* para fragmentos adecuados también es una tarea desafiante y esto conduce a una nueva dirección para futuros investigadores.

La distribución de índices multimedia a múltiples nodos permite la búsqueda sobre conjuntos de datos muy grandes, como es en el caso de imágenes y videos, sin embargo, presenta una serie de problemas, tal es el caso de determinar cómo distribuir documentos y consultas de manera efectiva entre los nodos para admitir consultas simultáneas y cómo resolver el problema en fallas en los nodos o caídas en la red. Un índice donde las particiones se basan en la distribución de vectores de características en el espacio original permite mejorar la redundancia y aumentar la eficiencia: los vecinos más cercanos solo están presentes en un número pequeño y establecido de particiones, reduciendo el número de nodos para resolver cada consulta; por lo tanto, André & João [28] estudiaron cómo los *hashes* dispersos ayudan a encontrar este equilibrio y crear mejores políticas de distribución para vectores de características de alta dimensión. Así mismo, se propuso *Distributed Indexing by Sparse-Hashing* (DISH), un índice *k*-nn distribuido para sistemas basados en la nube, fundamentado en *hashes* dispersos. La arquitectura de DISH se divide en dos tipos de nodos:

- 1) Nodos de coordinación, se encargan de gestionar la interacción entre los clientes y los nodos de índice. Reciben solicitudes de consulta de búsqueda, las enrutan a nodos de índice utilizando *hash* disperso y agregan los resultados de búsqueda recuperados.
- 2) Nodos de índice, que son responsables de almacenar fragmentos de índice. Reciben solicitudes de los nodos de coordinación para recuperar los documentos candidatos que existen en su fragmento de índice (es decir, un conjunto de listas de publicación asignadas a posiciones de *hashes* dispersos).

Las principales propiedades de DISH son:

1. Índice de similitud multimedia distribuida: política de asignación de documento a nodo que se basa en la similitud en algoritmos de *hash* dispersos bien fundamentados.
2. Fragmentación de índice robusta y redundante: cada fragmento inspeccionado aporta nuevos documentos candidatos relevantes a la clasificación. La indexación redundante

también contribuye a una degradación del rendimiento pequeña en caso de falla del nodo.

3. Asignación de recursos flexible y equilibrada: la cuantificación del valor de pertenencia de la partición del documento (es decir, el coeficiente *hash*) permite elegir qué nodos consultar (por ejemplo, no consultar nodos con valores de miembros bajos si la carga es demasiado alta).

Se concluyó que DISH es un algoritmo de fragmentación de índice de imágenes a gran escala mediante *hash* dispersos; el uso de *hashes* dispersos permite fragmentar documentos y consultas de manera equilibrada y redundante en todos los nodos; con una sobrecarga de fragmentación baja. DISH se encuentra conformado principalmente por el algoritmo K-SVD [29] (algoritmo para adaptar diccionarios con el fin de lograr representaciones de señales dispersas, que se basa en el algoritmo *k*-Means) que ofrece una vista ortogonal demasiado completa del espacio de funciones originales.

El almacenamiento de datos subcontratado demostró sus ventajas comerciales principalmente por su alta estabilidad, sin embargo, la seguridad de los datos confidenciales y la utilidad son dos requisitos esenciales en el almacenamiento de datos subcontratado. Una solución clásica es el cifrado que ayuda a resolver el problema de la seguridad de los datos, pero el descifrado de datos requiere mucho tiempo, donde la eficiencia de la consulta y la evaluación de datos se ve afectada como consecuencia. Yong-Feng et al. [30] propusieron un algoritmo memético distribuido (DMA) que permitió mejorar la privacidad y la utilidad de la base de datos, incluyendo un marco distribuido para mejorar la eficiencia en la optimización, un operador de recombinación dinámica para mejorar la búsqueda global, dos operadores de mutación diseñados para ayudar a organizar y crear elementos evolutivos y una SDLS (*splicing-driven local search*, búsqueda local impulsada por empalme) para mejorar la capacidad de búsqueda. Para lograr una mayor eficiencia de optimización en la fragmentación de bases de datos, se propone BBRDF (marco distribuido aleatorio balanceado). El proceso es el siguiente: la población del algoritmo se divide en múltiples subpoblaciones (SP). Cada SP evoluciona de forma independiente. Para lograr una comunicación efectiva entre las SP, se conectan según una topología predefinida. Con un intervalo dado, se realiza la migración. Antes de la migración, se

seleccionan individuos de élite de cada SP. Durante la migración, según la topología de comunicación, los individuos seleccionados de cada SP se envían a su SP vecina. Después de la migración, los individuos migrados se insertan en las SP correspondientes. Los individuos, así como la información sobre la evolución, se intercambian a través de la migración. Para mejorar el procedimiento de búsqueda global, se propuso un operador de recombinación de agrupación dinámica (DGR) para agregar y utilizar elementos evolutivos con el objetivo de extraer información valiosa sobre la fragmentación de los individuos progenitores existentes y generar individuos hijos con mayor aptitud. De igual manera se plantearon dos operadores de mutación, que son:

1. **Mutación de fusión:** al ejecutar la mutación de fusión, se eligen aleatoriamente dos fragmentos del esquema original. Luego, todos los elementos en estos dos fragmentos se combinan y dos fragmentos se fusionan.
2. **Mutación dividida:** se selecciona aleatoriamente un fragmento del esquema original. Cada grupo contiene parte de los elementos del fragmento original. Estos dos grupos se insertan en el nuevo individuo y actúan como dos fragmentos para reemplazar el fragmento elegido.

La ejecución del SDLS propuesto se divide en tres pasos. Primero, para cada individuo seleccionado, se enumeran sus requisitos de utilidad insatisfechos. Uno de los requisitos de utilidad se elige aleatoriamente para la construcción de fragmentos. Posteriormente, se marcan las posiciones de todos los elementos en el requisito de utilidad seleccionado. Como tercer paso, todos los elementos del individuo original se extraen de su fragmento actual y se combinan para construir un nuevo fragmento. El DMA propuesto abordó los problemas de fragmentación de bases de datos; los resultados experimentales mostraron un mejor rendimiento en comparación con otros enfoques de fragmentación, como es el caso del algoritmo heurístico.

Para garantizar la seguridad y sostenibilidad de los sistemas de energía eléctrica, los datos masivos de energía eléctrica tienen que procesarse y analizarse rápidamente para tomar decisiones en tiempo real. Las soluciones tradicionales suelen utilizar bases de datos relacionales para gestionar los datos de energía eléctrica; sin embargo, las bases de datos

relacionales no procesan y analizan eficientemente datos masivos de energía eléctrica. En [31] se presentó cómo se administran los datos de energía eléctrica utilizando HBase [32], una base de datos distribuida mantenida por Apache, no relacional y de código abierto que se ejecuta sobre HDFS (*Hadoop Distributed File System*, sistema de archivos distribuido de Hadoop) [33]; en un sistema que consta de clientes, monitores de estado, módulos de migración de datos y módulos de fragmentación de datos. El sistema gestiona los datos que provienen de varios PDAS (*Power Dispatching Automation System*, Sistema de automatización de despacho de energía) donde los PDAS garantizan la seguridad de los sistemas de energía eléctrica [34], además, constan de diversos clientes y una plataforma de procesamiento de datos. Los PDAS cargan los datos a través de clientes, luego los servidores usan la plataforma para almacenar y analizar datos. Diferentes PDAS utilizan distintos formatos para almacenar datos. Para resolver el problema del intercambio de datos se utilizó CIM/E, un estándar basado en el Modelo de Información Común (CIM) [35], que permite que el software de aplicación intercambie información sobre un sistema eléctrico de manera más eficiente. Al consultar los datos, el sistema analizará los datos consultados en archivos CIM/E y luego los enviará a los PDAS. Al almacenar los datos en HBase se considera la fragmentación de los datos a través de particiones de grafos de la red eléctrica. El monitor de estado recopila información de Entradas/Salida de disco, memoria, red y CPU en tiempo real de los servidores de la plataforma, y envía los datos al controlador de balance de carga de trabajo; en lugar de utilizar software de código abierto, el monitor de estado recopila el estado del servidor llamando a las APIs (*Application Programming Interfaces*, Interfaz de Programación de Aplicaciones) del sistema operativo. Para controlar el equilibrio de carga de trabajo, el sistema adopta un enfoque sencillo basado en el método de detección de umbral, de acuerdo al grado de desequilibrio de la carga de trabajo, cuando este grado alcanza el umbral, se iniciará con la migración de datos. Al momento de aplicar la migración de datos, las cargas de trabajo se encuentran desequilibradas, por lo cual, se abordó este problema calculando primero la cantidad de fragmentos de datos que se migran y generando posteriormente un plan de migración de datos correspondiente. Se comparó el sistema con otro sistema de almacenamiento de datos de PDAS basado en MySQL, en un clúster de 15 servidores, usando ocho conjuntos de datos de energía eléctrica cuyos tamaños varían de 1 a 128 millones de tuplas. Donde se realizó una consulta seleccionando la información del error de

la estación 1 emitida en enero de 2016. El sistema propuesto fue cuatro veces más rápido que el sistema basado en MySQL.

La seguridad se convierte en el principal problema que desalienta a las organizaciones a aprovechar los beneficios de la computación en la nube, porque los datos de los usuarios se almacenan en otros sitios y esto dificulta la confirmación de los diferentes niveles de seguridad de los datos. En [36] se presentó un nuevo mecanismo de seguridad que utiliza un método híbrido de algoritmos de cifrado y un sistema de distribución para mejorar la confidencialidad de la base de datos en la nube. De igual manera, se adoptó una técnica de fragmentación vertical del modelo de Alsirhani's et al. [37] para distribuir datos; sin embargo, el nuevo modelo mejoró el aislamiento en la nube utilizando la fragmentación híbrida en lugar de la fragmentación vertical. La investigación siguió una metodología de diseño mixto. Primero aplicaron estrategias cualitativas para diseñar el modelo propuesto en lenguaje de programación Java y usando el gestor de bases de datos Microsoft Access, donde: 1) Se ingresa el texto sin formato por parte del usuario; 2) Se realiza Cifrado/ Descifrado y fragmentación en el Proxy, y 3) Los datos encriptados se almacenan en la nube. La simulación de Alsirhani's et al. [37] también se desarrolló en Java y se utilizaron estrategias cuantitativas para probar el modelo (es decir, calcular y comparar las respuestas de las consultas en milisegundos). La evaluación del modelo propuesto implicó probar el esquema comparándolo con dos modelos (Alsirhani's y Popa's [37], [38]) en diferentes escalas, como el triángulo CIA (*Confidentiality, Integrity, and Availability*, Confidencialidad, integridad y disponibilidad), así como la complejidad y la distribución de datos a través de nubes esclavas. El modelo en [37] proporcionó una tasa de respuesta más rápida que el modelo propuesto, porque el segundo utiliza un procesamiento de consultas más complejo en la distribución de datos (híbrido), mientras que el modelo de Alsirhani's et al. [37] solo usa fragmentación vertical. Sin embargo, el enfoque propuesto brinda mejor seguridad porque los datos son ilegibles si algún intruso los obtiene después de la distribución. Finalmente, la técnica de fragmentación híbrida proporcionó distribución de datos a través de nubes esclavas, en primer lugar, mediante la implementación de la fragmentación vertical, seguida de la partición horizontal. La confidencialidad e integridad se aseguró mediante tres técnicas: con el modelo propuesto, la técnica distribuida segura [37] y la técnica de Popa's. La disponibilidad se logró

en el esquema propuesto y la técnica distribuida segura. La simplicidad de operación no caracterizó a ninguna de las tres técnicas. Si un atacante obtuviera los datos, estos serían ilegibles mediante la aplicación de la técnica de fragmentación híbrida, por consiguiente, los datos se dividen en pequeños fragmentos por ID de usuario en el modelo propuesto. La distribución de datos a través de la nube se logró mediante el modelo propuesto y la técnica de distribución segura a través de la implementación de la técnica de fragmentación.

En el Instituto Tecnológico de Orizaba se realizó un proyecto de recolección de datos históricos de este, donde se recuperó una gran cantidad de datos multimedia. Al final del proyecto se detectó que el aumento de contenido multimedia dificultaba la gestión óptima de estos datos; para resolver este problema, Rodríguez-Arauz et al. [39] diseñaron un sistema para el manejo de datos históricos del Instituto, el cual utilizó la fragmentación horizontal para optimizar las consultas basadas en contenido. El diseño del sistema de gestión de datos multimedia usó un método de fragmentación horizontal para optimizar las consultas basadas en contenido. Para la arquitectura del sistema se utilizó MVC (*Model-view-controller*, modelo-vista-controlador) por ser fácil de dar mantenimiento. MongoDB [40] se usó como sistema gestor de bases de datos por la velocidad que ofrece y por ser simple al realizar consultas basadas en contenido en la base de datos. JSF [41] se empleó por la flexibilidad para crear aplicaciones con el lenguaje Java puro, su facilidad de desarrollo de software y porque es gratuito. Debido al marco de desarrollo seleccionado, NetBeans [42] fue el IDE. UWE [43] se eligió como metodología de desarrollo porque se especializa en el desarrollo de aplicaciones Web y multimedia. Por último, se seleccionó BoofCV [44] porque es una biblioteca Java de código abierto que tiene facilidad de uso y alto rendimiento. El modelo está representado en los *beans* administrados que tienen acceso a los componentes de la interfaz pasando la información al modelo. Se utilizó BoofCV, para realizar la recuperación basada en contenido (imágenes) y Java Database Connectivity (JDBC) para controlar el acceso a MongoDB; aquí se aplicó el método de fragmentación horizontal propuesto en [45] porque considera consultas basadas en contenido. En la vista se utilizó XHTML (Lenguaje de marcado de hipertexto extensible), etiquetas de JSF, junto con las hojas de estilo (CSS) brindando al usuario una mejor visualización. En el controlador, el *servlet* JSF es el enlace entre el modelo y la vista; de acuerdo a la solicitud por el usuario, es la vista a

presentar. El proceso de la recuperación de imágenes basada en contenido muestra un formulario para ingresar la imagen del contenido a recuperar, luego el sistema evalúa si hay un archivo válido para hacer la recuperación basada en contenido. Una vez que se proporcionó un archivo válido, el sistema analiza la imagen y recupera el contenido más similar asignado en el tipo de fragmento específico de la propia imagen. Finalmente, se muestra una página de resultados donde el usuario es capaz de apreciar los resultados de la recuperación de la imagen basada en el contenido. La colección de imágenes está fragmentada considerando el tipo de atributo de acuerdo con Fasolin et al. [45], produciendo fragmentos horizontales nombrados: edificios, eventos, personas y equipos. El sistema mejoró las consultas basadas en contenido, al utilizar un método de fragmentación horizontal se obtuvieron como resultado menores costos de ejecución y tiempos de respuesta.

## 2.2 Análisis comparativo

La Tabla 2.1 presenta un análisis comparativo sobre los artículos relacionados con este proyecto, así como una breve descripción acerca de la problemática, contribución, tecnologías utilizadas, resultados obtenidos y el estado del artículo.

Tabla 2.1 Análisis comparativo de los trabajos relacionados

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
Castro-Medina et al. [8]	Traslape de fragmentos en un método de fragmentación, asignación y replicación de datos en la nube.	Aplicación Web llamada FRAGMENT, para la fragmentación, asignación y replicación de bases de datos en un entorno en la nube.	<ul style="list-style-type: none"> <li>• Arquitectura MVC.</li> <li>• Beans gestionados por JSF (JavaServer Faces).</li> <li>• AWS para implementar la aplicación.</li> <li>• NetBeans como IDE.</li> <li>• UWE como metodología de desarrollo.</li> <li>• MySQL como SGBD.</li> </ul>	Aplicación Web que implementa un método adecuado de fragmentación sobre sitios en la nube, realizando las operaciones de asignación y replicación automáticamente.	<p>Terminado</p> <p><b>Trabajos a futuro:</b></p> <p>Se propone agregar un modelo de fragmentación dinámica, basada en umbrales de costo para realizar fragmentaciones posteriores a la inicial.</p>

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
<p><b>Castro-Medina et al.</b> [14]</p>	<p>Traslape de fragmentos mediante la precedencia de costos obtenida de la MCRUD.</p>	<p>Algoritmo que utiliza la prioridad de costos al producir el esquema de fragmentación final. De este modo, la preservación de fragmentos se basa en la prioridad del costo y no en el tamaño.</p>	<ul style="list-style-type: none"> <li>• Arquitectura MVC.</li> <li>• Beans gestionados por JSF.</li> <li>• NetBeans como IDE.</li> <li>• UWE como metodología de desarrollo.</li> <li>• MySQL como SGBD.</li> </ul>	<p>Se adaptó el algoritmo al entorno de la aplicación Web FRAGMENT, mejorando el esquema producido al reducir el tiempo de ejecución de las consultas relacionadas con el atributo más costoso.</p>	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <p>Se utilizará la agrupación de fragmentos con el mismo costo para mejorar el esquema producido al evitar fragmentos con un solo elemento obtenido por operaciones de igualdad.</p>
<p><b>Castro-Medina et al.</b> [9]</p>	<p>El uso de las consultas basadas en contenido va en aumento, dado que los elementos multimedia a menudo se presentan dentro de las bases de datos. Por lo cual, las nuevas estrategias de fragmentación tienen que incluir este aspecto para proporcionar esquemas de mejor rendimiento.</p>	<p>Método de fragmentación horizontal dinámica para bases de datos multimedia.</p>	<ul style="list-style-type: none"> <li>• Aplicación Web XAMANA desarrollada en el lenguaje Java.</li> <li>• CBIR</li> <li>• Consulta k-NN y descriptor SURF.</li> <li>• Base de datos multimedia HITO.</li> </ul>	<p>El método de fragmentación dinámica desarrollado mejoró el rendimiento mediante la reducción del tiempo de ejecución de las operaciones, de forma precisa y adecuada.</p>	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <ol style="list-style-type: none"> <li>1. Se implementará un nuevo método para la fragmentación vertical considerando consultas basadas en contenido en XAMANA.</li> </ol>

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
					2. Se desarrollará un traductor de operaciones para la adaptación del código de programación a los nuevos esquemas producidos por este enfoque.
Safaei [5]	Dificultad al buscar y encontrar la imagen médica adecuada entre una gran cantidad de ellas; al usar la indexación de imágenes médicas basadas en contenido, las consultas resultan más complejas y consumen más tiempo que las basadas en texto.	<ol style="list-style-type: none"> <li>1. Investigación complementaria para la técnica de indexación de imágenes médicas que utiliza un enfoque más avanzado y flexible (es decir, la fragmentación híbrida en lugar de la fragmentación vertical pura).</li> <li>2. Técnica de indexación multidimensional basada en fragmentación híbrida para imágenes médicas para considerar la</li> </ol>	<ul style="list-style-type: none"> <li>• DICOM</li> <li>• Algoritmos de agrupamiento jerárquico.</li> <li>• Búsqueda o recuperación (<i>Fetch</i>).</li> <li>• Motor de búsqueda de código abierto Lucene.</li> <li>• Sistema operativo: Windows, máquina subyacente: Intel Core i5-4200 M @ 2.5.0 GHz RAM: 6 GB.</li> </ul>	La indexación multidimensional mejoró la precisión en la recuperación de información, en lugar de un solo índice se pasó a un conjunto de múltiples índices, considerando la relación semántica de los atributos de la imagen médica a través de la fragmentación híbrida (horizontal y vertical).	Finalizado  <b>Trabajos a futuro:</b>  Uso de los diferentes enfoques de fragmentación híbrida dinámica.

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
		perspectiva semántica de múltiples aspectos del contenido de la imagen médica.			
Schreiner et al. [16]	Falta de partición de datos en sistemas NewSQL.	Enfoque de partición híbrida para bases de datos NewSQL que permite al usuario definir las particiones de datos verticales y horizontales.	<ul style="list-style-type: none"> <li>• NewSQL incorpora y se basa en los conceptos y principios del lenguaje de consulta estructurado (SQL) y los lenguajes NoSQL (<i>Not only SQL</i>, No solo SQL).</li> <li>• Arquitectura VoltDB como base de datos relacional proveniente de NewSQL.</li> <li>• Algoritmo <i>hash</i> para implementar nuevos enfoques de partición vertical y, en consecuencia, híbrida.</li> <li>• OLTP-Bench, una <i>suite de benchmark</i> que permite realizar pruebas sobre diferentes sistemas de bases de datos comerciales.</li> <li>• Máquina servidor con cuatro procesadores Intel(R) Xeon(R) CPU E7-4850 (2,00 GHz)</li> </ul>	Hybrid-VoltDB, un enfoque para administrar la partición híbrida de datos materializado en VoltDB.	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <ol style="list-style-type: none"> <li>1. Evaluación del enfoque presentado con otros puntos de referencia y la ejecución de pruebas en un clúster distribuido físicamente.</li> <li>2. Desarrollo de un enfoque automatizado que considere la carga de trabajo para definir nuevos esquemas de partición para reducir la cantidad de transacciones distribuidas.</li> </ol>

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
			con 128 GB de RAM, ejecutando Linux 4.15.0-50 kernel (distribución Ubuntu 18.04.2 LTS). <ul style="list-style-type: none"> <li>• Docker2 para emular tres máquinas diferentes (<i>hosts</i>) conectadas por una red virtual.</li> <li>• Computadora externa con procesador Intel Core i5-2430M con 8 GB DDR3 1066 MHz RAM, 240 GB.</li> <li>• <i>Benchmark</i> TPC-C.</li> </ul>		
<b>Dahal &amp; Joshi</b> [22]	Dificultad en el diseño de bases de datos distribuidas al aplicar la fragmentación.	Encontrar el mejor algoritmo de fragmentación vertical.	<ul style="list-style-type: none"> <li>• Algoritmo de agrupamiento <i>k</i>-Means para crear los fragmentos.</li> <li>• Algoritmo basado en MST.</li> <li>• <i>Partition Comparator</i> herramienta utilizada para evaluar el costo total de acceso que se produce en los fragmentos de una relación.</li> </ul>	El algoritmo basado en KMC posee un mejor rendimiento en comparación con el algoritmo basado en MST.	Finalizado

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
<p><b>Noraziah et al.</b> [23]</p>	<p>Al implementar la replicación dentro de las bases de datos, el diseño de los esquemas se enfrenta a problemas de desperdicio de espacio de almacenamiento y aumento en el tiempo de procesamiento de las transacciones.</p>	<p>Algoritmo para manejar la replicación síncrona de bases de datos fragmentadas.</p>	<ul style="list-style-type: none"> <li>• Minería de datos, usando el algoritmo Apriori basado en reglas de asociación.</li> <li>• Método para la fragmentación.</li> </ul>	<p>BVAGQ-AR como algoritmo para la gestión de la replicación síncrona y las transacciones, capaz de preservar la consistencia del esquema en las bases de datos.</p>	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <p>Mejorar el algoritmo BVAGQ-AR, de tal manera que sea capaz de controlar todo tipo de fallas, como errores en la red, errores de software, errores de comunicación, entre otros.</p>

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
Ge et al. [24]	En la fragmentación dinámica, se tiene que considerar la preservación de la privacidad, así como el costo de comunicación durante la optimización.	Algoritmo de evolución diferencial distribuida basado en la transferencia de conocimiento (KT-DDE) para lograr el costo de comunicación óptimo y mantener la preservación de la privacidad.	<ul style="list-style-type: none"> <li>• DE-KT-Optimizer: variante que adopta los operadores de mutación y cruce basados en transferencia de conocimiento en KT-DDE.</li> <li>• DE-KT-Framework: en esta variante, se agrega el marco basado en la transferencia de conocimiento.</li> </ul>	Algoritmo KT-DDE que incluye un marco distribuido y un optimizador basado en DE.	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <ol style="list-style-type: none"> <li>1. Considerando la efectividad del algoritmo KT-DDE, se propone adaptar las estrategias en KT-DDE a otros problemas de fragmentación de bases de datos.</li> <li>2. Las estrategias basadas en la transferencia de conocimiento en KT-DDE se contemplarán en otros problemas de optimización dinámica.</li> </ol>

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
<p><b>Brendle et al.</b> [25]</p>	<p>Al trasladar las bases de datos a la nube, los proveedores de bases de datos como servicio tienen la responsabilidad de cumplir con los SLA, manteniendo los costos de hardware lo más bajo posible; sin embargo, la cantidad de espacio en memoria domina los costos de hardware.</p>	<p>SAHARA, un asesor que propone una partición de tablas para almacenes de columnas con un consumo mínimo de memoria y, al mismo tiempo, cumple con todos los SLAs de rendimiento.</p>	<ul style="list-style-type: none"> <li>• SAP HANA Cloud como base de datos como servicio.</li> <li>• Algoritmos de enumeración exactos y heurísticos para determinar un diseño de partición.</li> <li>• Hardware: CPU Intel Xeon E7-8870 v4 (4 núcleos) y 1 TB de DRAM.</li> <li>• Almacenamiento secundario: RAID de 8 HDD.</li> <li>• Cargas de trabajo: <i>Benchmark JCC-H, Join Order Benchmark (JOB)</i>.</li> </ul>	<p>Mejora a SAHARA que minimiza la huella de memoria y propone un nuevo modelo de costos. Integración de SAHARA en SAP HANA Cloud.</p>	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <p>Predecir la carga de trabajo en función de una carga de trabajo observada para decidir si la partición proactiva es beneficiosa. Por ejemplo, si los costos de repartición se amortizan mediante un mejor ajuste del diseño de la tabla.</p>
<p><b>Aggarwal et al.</b> [27]</p>	<p>En sistemas de bases de datos distribuidas, las bases de datos se encuentran en el mismo sistema, pero geográficamente están dispersas en múltiples ubicaciones; los usuarios acceden a los datos a través de sitios locales y remotos. El acceso a</p>	<p>Técnica para recuperar datos sin causar retrasos, aplicando el agrupamiento <math>k</math>-Medoids que se utiliza en la fragmentación vertical.</p>	<ul style="list-style-type: none"> <li>• <math>k</math>-Medoids como técnica de agrupamiento para crear fragmentos de la base de datos.</li> <li>• Matriz bidimensional (AUM).</li> <li>• Recuento de disimilitud (DC) para verificar la efectividad del algoritmo aplicado.</li> </ul>	<p>El algoritmo propuesto es simple en comparación con el árbol de expansión mínimo mejorado (EMST), ya que solo requiere una matriz AUM, usando el valor DC para la fragmentación y <math>k</math>-Medoids para</p>	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <p>Elegir <i>medoides</i> para fragmentos adecuados también es una tarea desafiante y esto conduce a una nueva dirección para futuras investigaciones.</p>

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
	sitios remotos genera retrasos, lo que provoca un rendimiento deficiente del sistema.			agrupar los atributos con la menor disimilitud.	
<p><b>Mourao &amp; Magalhaes</b> [28]</p>	<p>Dificultad al momento de distribuir índices a multimedia a múltiples nodos.</p>	<p>Esquema de fragmentación de índices, a través de <i>hashes</i> dispersos para fragmentar documentos y consultas de manera equilibrada y redundante en todos los nodos.</p>	<ul style="list-style-type: none"> <li>• Algoritmo KSVD que se encuentra incrustado en DISH.</li> </ul> <p><b>Para realizar las pruebas:</b></p> <ul style="list-style-type: none"> <li>• Máquinas: todos los nodos son instancias de máquinas virtuales (VM) en Microsoft Azure Standard, con cuatro núcleos virtuales (2,2 GHz Intel Xeon E5-2660), 28 GB de memoria y un SSD local de 200 GB.</li> <li>• Red: Los nodos están conectados a una red de 10 Gbps, con una latencia inferior a 1 ms entre nodos. La comunicación entre nodos se realiza a través de UDP (<i>User datagram protocol</i>, protocolo de datagramas de usuario).</li> </ul>	<p>DISH, un algoritmo de fragmentación de índices de imágenes a gran escala mediante <i>hashes</i> dispersos.</p>	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <p>El artículo se centró en un problema habitual y poco estudiado, relacionado con la indexación redundante de vectores multimedia de alta dimensión en múltiples computadoras. La novedad del problema abre muchas nuevas direcciones de investigación, como reducir la redundancia de datos mientras se mantiene la misma resiliencia ante fallas de nodos, o consultar menos</p>

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
			<ul style="list-style-type: none"> <li>Conjunto de datos: el conjunto de datos Billion Vectors [46], [47] usando un subconjunto de 100,000 de los vectores de entrenamiento provistos para calcular el diccionario K-SVD y los centroides de <math>k</math>-Medias.</li> </ul>		<p>nodos en situaciones de alta carga sobre escenarios.</p>
<p><b>Ge et al. [30]</b></p>	<p>Problemas de privacidad y utilidad en el almacenamiento subcontratado de datos.</p>	<p>Algoritmo memético distribuido (DMA) para la fragmentación de bases de datos subcontratadas.</p>	<ul style="list-style-type: none"> <li>Marco distribuido aleatorio balanceado (BBRDF).</li> <li>Operador DGR (<i>dynamic-grouping recombination</i>, recombinación de agrupación dinámica) para extraer información importante sobre la fragmentación.</li> <li>Operadores de mutación: mutación de fusión y mutación dividida.</li> <li>SDLS.</li> </ul>	<p>El algoritmo memético distribuido (DMA) incluye un marco distribuido para mejorar la eficiencia en la optimización en la fragmentación de las bases de datos, un operador de recombinación dinámica para mejorar la búsqueda global, dos operadores de mutación diseñados para ayudar a organizar</p>	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <p>Considerando la efectividad de los operadores propuestos en DMA, se aplicarán a otros problemas de optimización de ingeniería discreta. Además, dado que el marco distribuido propuesto es capaz de mejorar el intercambio de información entre</p>

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
				y crear elementos evolutivos y una SDLS para mejorar la capacidad de búsqueda.	los SP y mejorar la velocidad algorítmica, se utilizará en otros problemas de optimización de la propiedad compleja o de gran escala.
<b>Jin et al. [31]</b>	Las bases de datos relacionales no procesan y analizan eficientemente datos masivos de energía eléctrica cuando los datos van en aumento; por lo que se requiere del uso de una base de datos distribuida para solventar este problema.	Sistema que utiliza HBase para almacenar datos de energía eléctrica.	<ul style="list-style-type: none"> <li>• PDAS</li> <li>• CIM/E</li> <li>• HBase</li> <li>• API del sistema operativo.</li> </ul> <p><b>Para la evaluación:</b></p> <p>Se evaluó el sistema en un clúster de computadoras el cual contiene: 15 servidores Dawning CB60 con Intel de 2,60 GHz Xeon CPU E5-2670, memoria de 32 GB y disco SAS (<i>Serial Attached Small Computer System Interface</i>, interfaz pequeña conectada en serie de sistemas de cómputo) de 300 GB, que están conectados mediante conmutadores gigabit.</p>	La incorporación de HBase en el sistema mejoró la administración de datos; ya que el sistema se encuentra conformado por clientes, monitores de estados, módulos de migración de datos y módulos de fragmentación de datos que se basa en la partición de grafos; el gran volumen de datos provocaba una carga masiva.	Finalizado

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
			<p>Con un Sistema operativo Red Hat Linux 6.2, Hadoop 1.0.4 y HBase 0.94.6.1</p> <p><b>Para realizar la comparación con el sistema existente:</b></p> <ul style="list-style-type: none"> <li>Se comparó el sistema con otro sistema de almacenamiento de datos PDAS basado en MySQL con un clúster de 15 servidores.</li> </ul>		
<p><b>Awad et al.</b> [36]</p>	<p>La seguridad se convierte en el principal problema que desalienta a las organizaciones a aprovechar los beneficios de la computación en la nube.</p>	<p>Mecanismo de seguridad que utiliza un método híbrido de algoritmos de cifrado y un sistema de distribución para mejorar la confidencialidad de la base de datos en la nube.</p>	<ul style="list-style-type: none"> <li>Lenguaje Java para la construcción del modelo, como también para la simulación del modelo de Alsirhani's.</li> <li>Base de datos Microsoft Access.</li> </ul>	<p>Modelo que se implementó utilizando Java, adoptando una técnica de fragmentación híbrida, que distribuye la base de datos en la nube, empleando las vistas del proveedor y el nivel de confidencialidad.</p>	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <ol style="list-style-type: none"> <li>Investigar cómo se reducirá el número de columnas usando técnicas híbridas de encriptación, en un método que proporciona varios tipos de consultas.</li> </ol>

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
					2. Como segunda dirección, explorar la manera de mejorar la distribución de datos en la nube y garantizar la integridad aumentando la comunicación entre diferentes bases de datos en la nube utilizando un canal seguro.
<p><b>Rodríguez-Arauz et al.</b> [39]</p>	<p>Al finalizar el proyecto que recolecta los datos históricos del Instituto Tecnológico de Orizaba, se detectó que a causa del aumento del contenido multimedia es difícil la gestión de los datos.</p>	<p>Sistema de gestión de datos históricos del Instituto Tecnológico de Orizaba, el cual utiliza la fragmentación horizontal para optimizar las consultas basadas en contenido.</p>	<ul style="list-style-type: none"> <li>• Para la arquitectura del sistema se utilizó MVC.</li> <li>• MongoDB como sistema de base de datos.</li> <li>• <i>JavaServer Faces</i>.</li> <li>• NetBeans como IDE.</li> <li>• UWE como metodología de desarrollo.</li> <li>• BoofCV para realizar la recuperación basada en contenido (imágenes).</li> </ul>	<p>El sistema mejoró las consultas basadas en contenido, al utilizar un método de fragmentación horizontal se obtuvieron como resultado menores costos y tiempos de ejecución.</p>	<p>Finalizado</p> <p><b>Trabajos a futuro:</b></p> <p>1) Implementación del sistema utilizando las tecnologías seleccionadas y la incorporación de un método de</p>

Artículo	Problema	Contribución	Tecnologías empleadas	Resultado	Estado
			<ul style="list-style-type: none"><li>• JDBC para controlar el acceso a MongoDB.</li><li>• XHTML.</li><li>• CSS.</li></ul>		<p>fragmentación vertical para una mejor optimización de las consultas basadas en contenido.</p> <p>2) Se incluirán consultas complejas conjuntivas para la evaluación del esquema de fragmentación horizontal.</p>

Con el auge de la tecnología, hoy en día, todo tipo de aplicación, sistema y otro tipo de *software*, hace uso de las bases de datos para almacenar grandes volúmenes de información. En el caso de los datos multimedia, implica que aumenten los costos de transmisión de datos y los tiempos de respuesta de las consultas en las bases de datos multimedia.

Los artículos relacionados con esta investigación reportan el uso de técnicas de fragmentación, sin embargo, no todos consideran datos multimedia y la fragmentación híbrida dinámica.

La fragmentación horizontal se incluye en los trabajos [8], [9], [14], [39]. Castro-Medina et al. [8] implementaron un método de fragmentación sobre sitios en la nube, realizando las operaciones de asignación y replicación automáticamente en la aplicación Web FRAGMENT. Posteriormente, en [14] se añadió a esta misma aplicación un algoritmo que utiliza la prioridad de costos al producir esquemas de fragmentación mejorando el desempeño al reducir el tiempo de ejecución de las consultas relacionadas con el atributo más costoso. En [9] se propuso un método de fragmentación horizontal dinámica para bases de datos multimedia implementado en XAMANA, que mejoró el rendimiento mediante la reducción del tiempo de ejecución de las operaciones, de forma precisa y adecuada. Así como también, en [39] el sistema de gestión de datos históricos del Instituto Tecnológico de Orizaba utiliza la fragmentación horizontal para optimizar las consultas basadas en contenido, teniendo como resultado menores costos de ejecución.

En contraste, [22], [24], [25], [27], [28], [30] incorporan la fragmentación vertical. En [22] se encontró que el mejor algoritmo para la fragmentación vertical es  $k$ -Means Clustering (KMC). En [24] al aplicar la fragmentación dinámica, también fue necesario proteger la privacidad de las bases de datos distribuidas, para solucionar este problema, se propuso un algoritmo de evolución diferencial distribuida basado en la transferencia de conocimiento (KT-DDE) para lograr el costo de comunicación óptimo y mantener la preservación de la privacidad. En [25] se presentó SAHARA, un asesor que propone una partición de tablas para almacenes de columnas con un consumo mínimo de memoria; integrado en un prototipo en una base de datos en la nube. En [27] se presentó la técnica para recuperar datos aplicando el agrupamiento  $k$ -Medoids, teniendo buenos resultados. En [28] la distribución de índices multimedia a múltiples nodos fue

complicada, por tal motivo, se presentó DISH, un algoritmo de fragmentación de índice de imágenes a gran escala, adaptando  $k$ -Means para calcular los *hashes* dispersos. En [30] a través del algoritmo memético distribuido (DMA) se mejoró la eficiencia en la optimización en la fragmentación de las bases de datos.

La fragmentación híbrida se abordó en [5], [16], [23], [36]. En el primero se abordó el problema de la indexación al momento de aplicar la fragmentación de imágenes, ya que resultaba compleja la búsqueda de imágenes en una gran cantidad; como segundo, se presentó un enfoque de partición híbrida para bases de datos NewSQL que permite al usuario definir las particiones de datos verticales y horizontales. En el tercer artículo, para resolver el problema de la replicación dentro de las bases de datos, se implementó el algoritmo de BVAGQ-AR para la gestión de la replicación síncrona de la base de datos fragmentada y las transacciones; y en el cuarto, al aplicar la fragmentación sobre sitios en la nube, la seguridad fue fundamental, para solucionar el problema se implementó un modelo adoptando una técnica de fragmentación híbrida, que distribuye la base de datos en la nube, mejorando la confidencialidad de los datos.

En [31] se usaron bases de datos distribuidas como HBase permitiendo almacenar los datos masivos de energía del sistema de control de una red eléctrica, además, se adoptó una estrategia de fragmentación de datos basada en la partición de grafos.

El objetivo de este proyecto es extender la aplicación Web XAMANA [9], con un nuevo método para la fragmentación híbrida dinámica para bases de datos multimedia.

### 2.3 Propuesta de solución

Para llegar a un resultado satisfactorio, se proponen una serie de pasos que dividen el trabajo en etapas, las cuales estarán sujetas a la metodología de desarrollo elegida, como se muestra en la Figura 2.1.

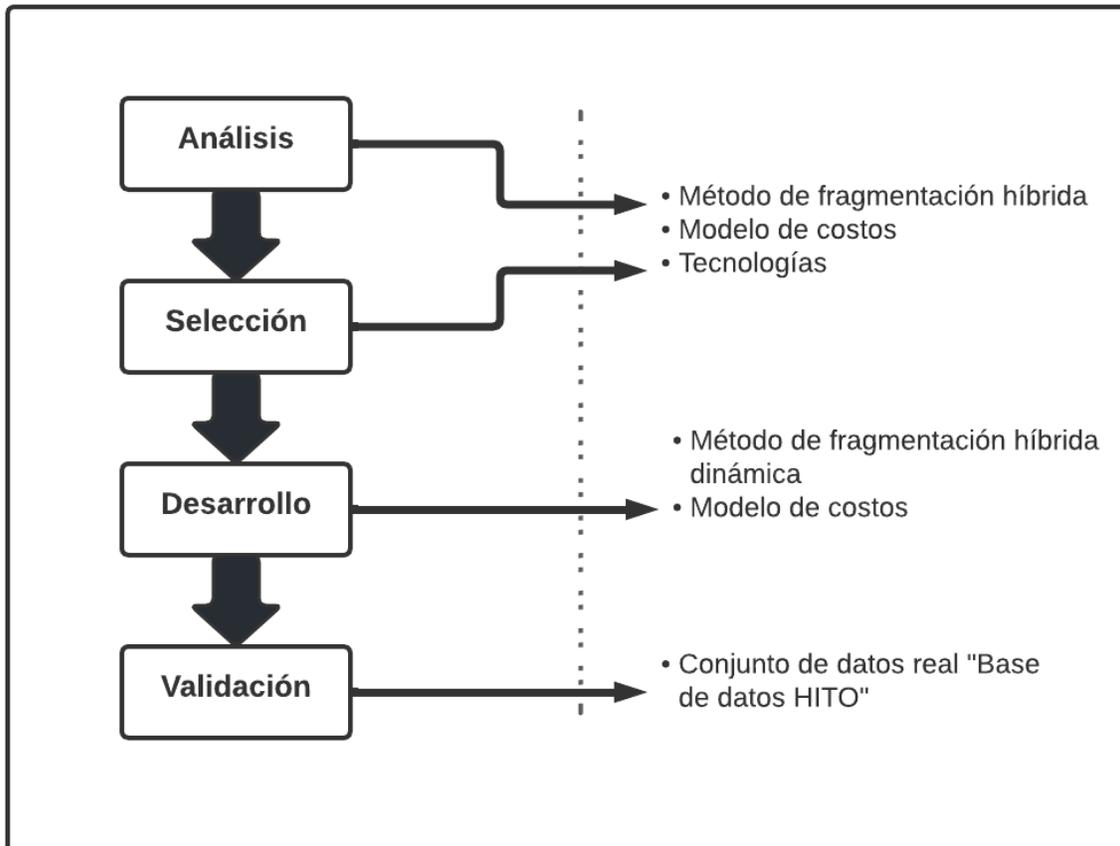


Figura 2.1 Estructura de la propuesta de solución

En la Figura 2.1 se observan los pasos propuestos para la obtención de la solución, los cuales se describen a continuación.

1. **Análisis:** En esta primera etapa se analizarán diferentes artículos por medio de la búsqueda en las principales bibliotecas digitales como ACM Digital Library, SpringerLink, ScienceDirect e IEEE Xplore, de los métodos de fragmentación híbrida estáticos y dinámicos, para bases de datos multimedia, de igual manera, los modelos de

costos utilizados para evaluar esquemas de fragmentación; además, las tecnologías a utilizar para el desarrollo.

2. **Selección:** Se seleccionará un método de fragmentación híbrida, así como un modelo de costos, de los trabajos anteriormente analizados; además de incluir las tecnologías. La técnica elegida será aquella que cumpla con los siguientes criterios: tener una solución completa, que considere la fragmentación híbrida dinámica, que tome en cuenta datos multimedia, que incluya un modelo de costos, fácil de implementar y que se enfoque en mejorar el desempeño de las consultas.
3. **Desarrollo:** En esta tercera etapa se desarrollará el método de fragmentación híbrida dinámica y el modelo de costos utilizando las tecnologías seleccionadas en la fase anterior.
4. **Validación:** Finalmente, se comparará el método de fragmentación y el modelo de costos desarrollado con el trabajo seleccionado del análisis comparativo, por medio de la base de datos HITO. De esta forma se comprobará la efectividad de la solución propuesta.

A continuación, se muestra la Tabla 2.2, que presenta la alternativa de solución.

Tabla 2.2 Alternativa de solución

Aspecto	Propuesta
Lenguaje de Programación	Java
Marco de Trabajo	JSF
IDE	NetBeans
Metodología	UWE
SGBD	MySQL y MongoDB

Java es un lenguaje de programación y una plataforma informática que surgió en el año de 1995 por Sun Microsystems. Desde su inicio, Java evolucionó logrando impulsar gran parte del mundo digital, proporcionando una plataforma confiable sobre la que se construyen diversos servicios y aplicaciones, tal es el caso de, aplicaciones móviles en Android, aplicaciones de escritorio/Web, servidores Web y servidores de aplicación, entre otros. Además, es un lenguaje orientado a objetos que provee una estructura clara en los programas y permite reutilizar el

código, reduciendo los costos de desarrollo; y de código libre, logrando que cualquier individuo lo adquiera de manera gratuita. Java está diseñado para el desarrollo de aplicaciones portables de alto rendimiento para el más amplio rango de plataformas informáticas. Las aplicaciones en entornos heterogéneos permiten estar a disposición de todo el mundo, por lo tanto, las empresas proporcionan más servicios mejorando la productividad, las comunicaciones y colaboración del usuario final, reduciendo el costo tanto para aplicaciones de usuario, como de empresa [48].

JavaServer Faces es una tecnología y marco estándar de interfaz de usuario (UI) orientado a componentes para la plataforma Java EE (*Enterprise Edition*, Edición Empresarial), en términos más comunes es un marco Web basado en Java. Además, JSF se basa en el marco de trabajo MVC basado en las API de *Servlets* que proporciona un conjunto de componentes de etiquetas definidas en páginas XHTML mediante *Facelets*. *Facelets* se define como un elemento fundamental de JSF que proporciona características de plantillas y de creación de componentes compuestos. JSF usa las páginas *Facelets* como vistas, objetos JavaBeans como modelos y métodos de objetos como controladores. El *servlet FacesServlet* realiza toda la tarea compleja de procesar las peticiones HTTP, obtener los datos de entrada, validarlos y convertirlos, colocarlos en los objetos del modelo, invocar las acciones del controlador y renderizar la respuesta utilizando el árbol de componentes.

Es importante mencionar que JSF está integrado en cualquier servidor de aplicaciones Java EE compatible, como WebLogic de Oracle y GlassFish *Open Source Edition* (Edición de Código Abierto), pero también permite utilizarse como una biblioteca independiente de contenedores de *servlets* como Tomcat y Jetty [49], [50].

NetBeans IDE es un entorno de desarrollo integrado de código abierto y gratuito, que se encuentra disponible para los sistemas operativos Windows, Mac, Linux y Solaris. NetBeans se utiliza para el desarrollo de aplicaciones, proporcionando soporte gratuito y completo para tecnologías y plataformas de Oracle; incluye modularidad al código, es decir, permite que las aplicaciones se desarrollen en módulos (como componente de un software). El IDE ofrece herramientas de primer nivel y simplifica el desarrollo de aplicaciones Web, empresariales, de

escritorio y móviles que utilizan las plataformas Java y HTML5. Además, ofrece soporte para el desarrollo de aplicaciones PHP y C/C++.

También, NetBeans facilita la gestión de bases de datos de tipo objeto-relacional ORDBMS (*Object-Relational Data Base Management System*, Sistema de Gestión de Base de Datos Objeto-Relacional); accediendo directamente desde el IDE. El IDE admite conexiones OCI (*Oracle Cloud Infrastructure*, Infraestructura en la nube de Oracle) y JDBC a Oracle Database. Las funciones de acceso completo a los datos están listas para usar, como la capacidad de leer, crear, actualizar y eliminar datos directamente dentro del IDE, con el apoyo de un editor SQL repleto de funciones [51].

UWE es una metodología de ingeniería que especifica de mejor manera una aplicación Web en su proceso de creación, el cual aprovecha la notación estándar del UML (*Unified Modeling Language*, Lenguaje de modelado unificado) [52], e incorpora elementos que son propios del desarrollo Web; que ayuda al ingeniero Web en las diferentes fases del ciclo de vida del desarrollo. UWE proporciona una mayor accesibilidad y manejabilidad de los requisitos por la inclusión en las primeras etapas del proceso de desarrollo [53].

En su implementación se contemplan las siguientes etapas y modelos [54]:

- 1. Análisis de requisitos.** Plasma los requisitos funcionales de la aplicación Web mediante un modelo de casos de uso.
- 2. Modelo de contenido.** Es el modelo conceptual del dominio de aplicación tomando en cuenta los requerimientos especificados en los casos de uso, y se define, mediante un diagrama de clases, los conceptos a detalle involucrados en la aplicación.
- 3. Modelo de navegación.** Basado en el análisis de requisitos y el modelo de contenido, se obtiene el modelo de navegación; se representa con clases de navegación de los objetos dentro de la aplicación y un conjunto de estructuras como son índices, menús y consultas.
- 4. Modelo de presentación.** Basado en el modelo de navegación y en los aspectos de la interfaz de usuario (requisitos), se obtiene el modelo de presentación; donde se representan las interfaces de usuario por medio de vistas abstractas.

- 5. Modelo de proceso.** Representa el aspecto que tienen las actividades que se conectan con cada clase de proceso.

Como se hace notar, UWE provee diferentes modelos que describen una aplicación Web desde varios puntos de vista abstractos. Cada uno de estos modelos se representan como paquetes UML, dichos paquetes son procesos relacionados que se refinan en iteraciones sucesivas durante el desarrollo de UWE [43].

MySQL [55] es el sistema de administración de bases de datos SQL de código abierto más usado, desarrollado, distribuido y respaldado por Oracle Corporation. Se considera de código abierto porque se descarga de su página oficial de forma gratuita, además, soporta la elaboración de nuevas funciones en colaboración con diferentes usuarios, y admite cambiar el código fuente para adaptarlo a las necesidades del usuario. MySQL es un gestor de bases de datos relacional que almacena los datos en estructura de tablas separadas. La parte de SQL de MySQL significa que utiliza el lenguaje de consulta estructurado, que consiste en la definición, manipulación e integridad de la información representada por los datos que se almacenan en las bases de datos.

MySQL es la principal opción de SGBD, por consiguiente, hace que sea compatible con diversas aplicaciones y lenguajes de programación. MySQL está presente en aplicaciones basadas en Web que constantemente están en crecimiento, como son: Facebook, Twitter, Booking, entre otros. Además, es una opción popular como base de datos integrada, distribuida por miles de ISV (*Independent Software Vendor*, Vendedor Independiente Software) y OEM (*Original Equipment Manufacturer*, Fabricante de Equipos Originales).

MongoDB [56] es una base de datos de documentos que ofrece escalabilidad y flexibilidad, y un modelo de consultas e indexación avanzado; que se encuentra disponible en la nube o como servidor. El modelo de documentos de MongoDB resulta muy fácil de aprender y usar, y proporciona a los desarrolladores todas las funcionalidades que necesitan para satisfacer los requisitos más complejos a cualquier escala. Se proveen controladores para más de diez lenguajes, y la comunidad ofrece varias decenas más.

MongoDB almacena datos en documentos flexibles similares a JSON (*JavaScript Object Notation*, Notación de Objetos de JavaScript), por lo que, los campos varían entre documentos y la estructura de datos es capaz de cambiar con el tiempo.

## Capítulo 3. Aplicación de la metodología

En este capítulo se aborda la aplicación de la metodología seleccionada para la realización del método de fragmentación híbrida dinámica para bases de datos multimedia y se divide en dos partes. La primera muestra los resultados de una búsqueda y un análisis exhaustivo de los métodos de fragmentación híbrida dinámica para seleccionar una técnica que cumpla con ciertos criterios. La segunda describe el diseño de la aplicación Web para realizar la ejecución de la técnica seleccionada. Además, se aborda también la especificación de la técnica seleccionada y la manera en la que se desarrolló la fragmentación híbrida dinámica.

### 3.1 Análisis

Con base en el objetivo general y específico, se realizó un análisis de los trabajos relacionados con la fragmentación híbrida dinámica utilizando la metodología propuesta en la Figura 3.1.

La Figura 3.1 muestra el proceso usado para la selección y análisis de cada artículo. Para llevar a cabo esto, primero se realizó una búsqueda en las principales bibliotecas digitales: 1) ACM Digital Library, 2) IEEE Xplore Digital Library, 3) ScienceDirect (Elsevier), y 4) SpringerLink, a partir del año 2010 hasta 2022. El segundo paso indica que se utilizó una búsqueda basada en palabras claves (*keywords*) las cuales fueron: 1) *Hybrid Fragmentation* (Fragmentación Híbrida), 2) *Mixed Fragmentation* (Fragmentación Mixta), 3) *Hybrid Partitioning* (Partición Híbrida), y 4) *Mixed Partitioning* (Partición Mixta).

Las publicaciones que no reunieron estas características y no eran adecuadas para el estudio fueron descartadas de acuerdo con el siguiente criterio establecido: Tesis de maestría o doctorado, libros y artículos que no estén escritos en inglés.

El proceso de selección resultó en 34 artículos, los trabajos se agruparon por editorial y año, posteriormente, se analizaron por siete características. Las propiedades consideradas fueron 1) *Benchmark*: Base de datos estándar usada para la validación del método, 2) *Compleitud*: Si el artículo tiene toda la información necesaria para implementar el método, 3) *Fácil de implementar*: El método no requiere conocimientos especializados para desarrollarlo e

implementarlo, 4) Modelo de costos: Si el método usa un modelo de costos para determinar el esquema de la fragmentación híbrida, 5) Dinámica o estática: El método es capaz de adaptar el esquema de acuerdo con los cambios en los patrones de acceso a la base de datos, o los elementos (atributos y/o tuplas) se asignan a un fragmento solo una vez en el momento de la creación, y sus ubicaciones nunca cambian, 6) Tipo de repositorio: Tipo de almacenamiento que se utilizó, y 7) Consultas basadas en contenido: Si el método considera la recuperación de objetos multimedia mediante contenido. Además, los artículos también se agruparon por editorial y año de publicación.

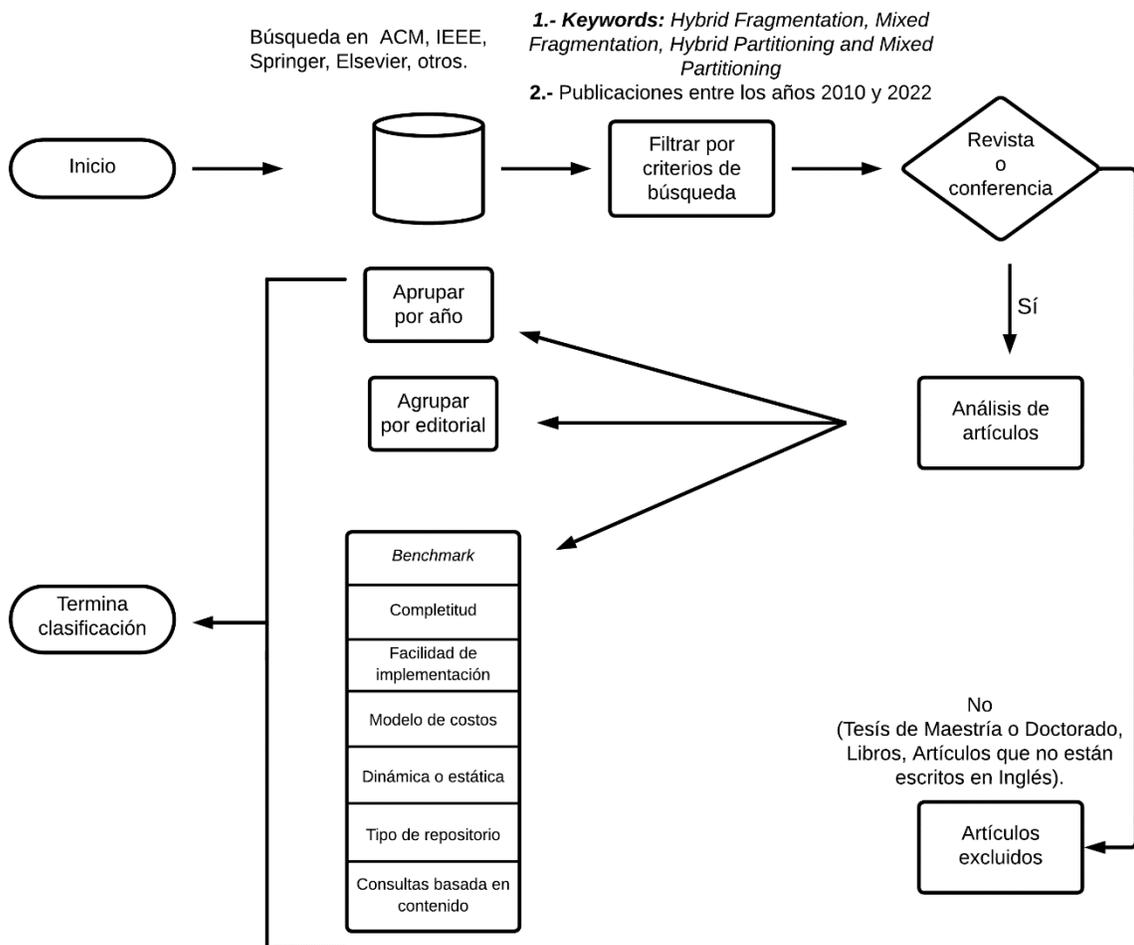


Figura 3.1 Metodología de búsqueda y evaluación de los trabajos relacionados

La Tabla 3.1 describe el registro de los artículos y compara todos los trabajos encontrados en las bibliotecas digitales.

Tabla 3.1 Comparación de los trabajos relacionados

Artículo	Benchmark	Compleitud	Fácil de implementar	Modelo de costos	Dinámica o estática	Tipo de repositorio	Consultas basadas en contenido
Chbeir et al.[57]	No se utilizó	Sí	No	No	Estática	Multimedia	Sí
Kling et al. [58]	XPathMark	Sí	No	Sí	Estática	XML	No
Jindal & Dittrich [59]	TPC-H ( <i>Transaction Processing Performance Council</i> , Consejo de Rendimiento del Procesamiento de Transacciones) y SSB ( <i>Star Schema Benchmark</i> , Punto de Referencia de Esquema de Estrella)	Sí	Sí	Sí	Dinámica	Relacional y Data Warehouse	No
Gorla et al.[60]	No se utilizó	Sí	No	Sí	Estática	Relacional	No
Song & Chen [61]	SSB	Sí	Sí	Sí	Estática	Relacional	No
Chen et al. [62]	RUBiS and TPC-W	Sí	No	Sí	Estática	Relacional	No
Wang et al. [63]	TPC-C	Sí	Sí	Sí	Dinámica	Relacional	No
Kechar & Nait Bahloul [64]	XML Data Warehouse Benchmark (XWB)	Sí	No	No	Estática	XML	No
Chen et al. [65]	No se utilizó	Sí	No	Sí	Dinámica	Objecto-Relacional	No
Harikumar & Ramachandran [66]	No se utilizó	Sí	No	No	Estática	Relacional-Distribuida	No
Al-Kateb et al. [67]	TPC-H	No	No	Sí	Estática	Multidimensional	No

Artículo	Benchmark	Compleitud	Fácil de implementar	Modelo de costos	Dinámica o estática	Tipo de repositorio	Consultas basadas en contenido
Rodríguez-Mazahua et al. [68]	No se utilizó	Sí	Sí	Sí	Estática	Multimedia	No
Padiya et al. [69]	TPC-C	Sí	Sí	Sí	Estática	Objecto-Relacional	No
Sun et al. [70]	Big Data Benchmark y TPC-H	No	No	Sí	Estática	NoSQL	No
Rani et al. [71]	Forest CoverType dataset	Sí	Sí	No	Estática y Dinámica	Relacional-Distribuida	No
Mourão & Magalhães [72]	No se utilizó	No	No	No	Estática	Multimedia	Sí
Durand et al. [73]	TPC-H	No	No	Sí	Estática	Relacional-Distribuida en la nube	No
Koong et al. [74]	No se utilizó	No	Sí	No	Estática	XML	No
Schreiner et al. [75]	TPC-W, and YCSB (Yahoo! Cloud Serving Benchmark, Yahoo! punto de referencia de servicio en la nube)	No	No	No	Estática	NewSQL	No
Vogt et al. [76]	No se utilizó	No	Sí	Sí	Dinámica	Polystorage Distributed	No
Chawla et al. [77]	The Lehigh University Benchmark (LUBM)	Sí	No	No	Estática	RDF Data Warehouse	No
Awad et al. [36]	No se utilizó	No	Sí	No	Estática	Relacional	No
Schreiner et al. [16]	TPC-C	Sí	Sí	No	Estática	NewSQL	No
Badran et al. [78]	No se utilizó	No	No	No	Estática	Relacional	No
Pinnecke et al. [79]	TPC-C	No	No	Sí	Estática	Relacional y Data Warehouse	No

<b>Artículo</b>	<b>Benchmark</b>	<b>Compleitud</b>	<b>Fácil de implementar</b>	<b>Modelo de costos</b>	<b>Dinámica o estática</b>	<b>Tipo de repositorio</b>	<b>Consultas basadas en contenido</b>
Kulba & Somov. [80]	No se utilizó	No	No	Sí	Dinámica	Distribuida	No
Noraziah et al. [23]	No se utilizó	Sí	No	No	Dinámica	Relacional	No
Kang et al. [81]	HAP ( <i>Hybrid Access Patterns</i> , Patrones de acceso híbridos) y TPC-H	Sí	No	Sí	Dinámica	Relacional	No
Badran et al. [82]	No se utilizó	Sí	Sí	No	Estática	Relacional	No
Azila et al. [83]	No se utilizó	No	Sí	No	Estática	Relacional-Distribuida	No
Patel et al. [84]	No se utilizó	Sí	Sí	Sí	Estática	Multimedia	No
Ahmed & Alluhaibi [85]	No se utilizó	Sí	No	Sí	Estática	Archivos	No
Safaei [5]	No se utilizó	Sí	Sí	No	Estática	Multimedia	No
Cantini et al. [86]	No se utilizó	Sí	No	No	Estática	Archivos	No

Los artículos seleccionados fueron evaluados utilizando la metodología mostrada en la Figura 3.1 para determinar si tienen las características deseadas. Algunos de estos cumplen con la mayoría de las cualidades, pero no toman en cuenta datos multimedia: [59], [63], mientras que, [68] y [84] sí consideran datos multimedia, pero no la fragmentación dinámica.

En [68] se propuso un método de partición híbrida para bases de datos multimedia que considera el tamaño de los atributos y la selectividad de los predicados para generar esquemas de fragmentación estática, el modelo de costos se conformó por dos partes, costo de acceso a datos irrelevantes y costo de transporte, lo que permitió reducir el tiempo de respuesta al momento de realizar consultas. Por otra parte, en la técnica de partición de datos sin procesar WSAC (*Workload and Storage Aware Cost-based*, basada en costos y consciente del almacenamiento y de la carga de trabajo) propuesta en [84], el modelo de costos extrae cada atributo del archivo de datos sin procesar y lo carga en el sistema de bases de datos para encontrar el tiempo de extracción, de carga y el tamaño del atributo; los valores obtenidos son almacenados en una lista, por lo cual, la técnica mejora el tiempo de procesamiento de consultas frecuentes.

### 3.2 Selección

De los 34 artículos analizados en la sección 3.1 Análisis, solo cuatro trabajos ([59], [63], [68], [84]) cumplen con la mayoría de los criterios requeridos para esta investigación tales como: completitud, fácil de implementar, involucran un modelo de costos y en los dos primeros se propone un método de fragmentación dinámica, es decir, es capaz de adaptarse a nuevas cargas de trabajo; sin embargo, una vez terminado de evaluar se descartaron los dos primeros por no considerar datos multimedia. En [59] los autores propusieron un almacén de datos autoajutable (AutoStore), permitiendo que se adapte el esquema de partición híbrida (Horizontal- Vertical) a las cargas de trabajo, sin necesidad de que el DBA (*Database Administrator*, Administrador de Bases de Datos) intervenga al momento de fragmentar. El algoritmo  $O^2P$  (*One-dimensional Online Partitioning*, Partición en Línea-unidimensional) estima el costo por medio de un optimizador basado en costos existentes; por otra parte, [63] resolvieron el problema de la distribución de datos con el modelo de triangulo DaWN (*Data, Worload and Nodes*, Datos, Carga de Trabajo y Nodos), el cual, utiliza datos, la carga de trabajo y los nodos que permiten

realizar la partición de manera horizontal-vertical, basándose en el modelo de costos como eje central para mejorar el desempeño, con la arquitectura ADDS (*Automatic Data Distribution Solution*, Solución de Distribución Automática de Datos) para aplicaciones OLTP, capaz de adaptarse a nuevas cargas de trabajo.

Por lo tanto, ambos trabajos no consideraron datos multimedia, además en Wang et al. [63] el equilibrio de la carga de trabajo y la replicación son dos problemas que no se abordaron en su propuesta, por tal motivo, estos dos artículos se descartaron para su comparación con el método propuesto.

El artículo seleccionado para su evaluación es el de Rodríguez-Mazahua et al. [68], ya que al igual que [84] el método de fragmentación híbrida propuesto no considera la parte dinámica, pero contempla datos multimedia y el modelo de costos se enfoca en considerar el acceso a datos irrelevantes y el costo de transporte, esto hace que sea un trabajo adecuado para la comparación.

### 3.3 Desarrollo

En esta sección se presenta el diseño de la aplicación Web, diagramas y procedimientos siguiendo la metodología establecida en la sección 2.3 Propuesta de solución, que permitirá el desarrollo del método de fragmentación híbrida.

En la Figura 3.2 se presenta el flujo de trabajo para el método de la fragmentación híbrida dinámica para bases de datos multimedia.

El primer paso es obtener el esquema de fragmentación horizontal y vertical. Posteriormente, se crea el esquema horizontal-vertical respetando asignaciones horizontales, así mismo, se genera el esquema vertical-horizontal considerando asignaciones verticales. Después de determinar ambos tipos de fragmentación híbrida, se comparan y se elige el esquema que tenga el menor costo para aplicarlo en la base de datos multimedia.

Al aplicar el esquema, internamente se inicia el análisis permanente de *logs* (la parte dinámica de la fragmentación), el cual se detendrá cuando el DBA lo considere necesario.

Para llevar a cabo la fragmentación dinámica se realizan los siguientes pasos:

1. Se evalúa el umbral de operaciones con un valor inicial que el DBA considere apropiado, de no ser alcanzado el umbral, no se realiza ningún cambio, en caso contrario, se inicia el paso dos.
2. Análisis de costos del esquema actual: se llevan a cabo las operaciones necesarias para obtener costos del esquema actual y continuar con el paso tres.
3. Se evalúa el umbral de desempeño, si este umbral no es sobrepasado por los resultados obtenidos del análisis de costos, no se realiza ningún cambio, en caso contrario, se inicia el paso cuatro.
4. Se realiza una re-fragmentación y asignación de fragmentos sobre el esquema actual.
5. Por último, se actualizan los vigilantes con el nuevo esquema en los sitios y se regresa al primer paso.

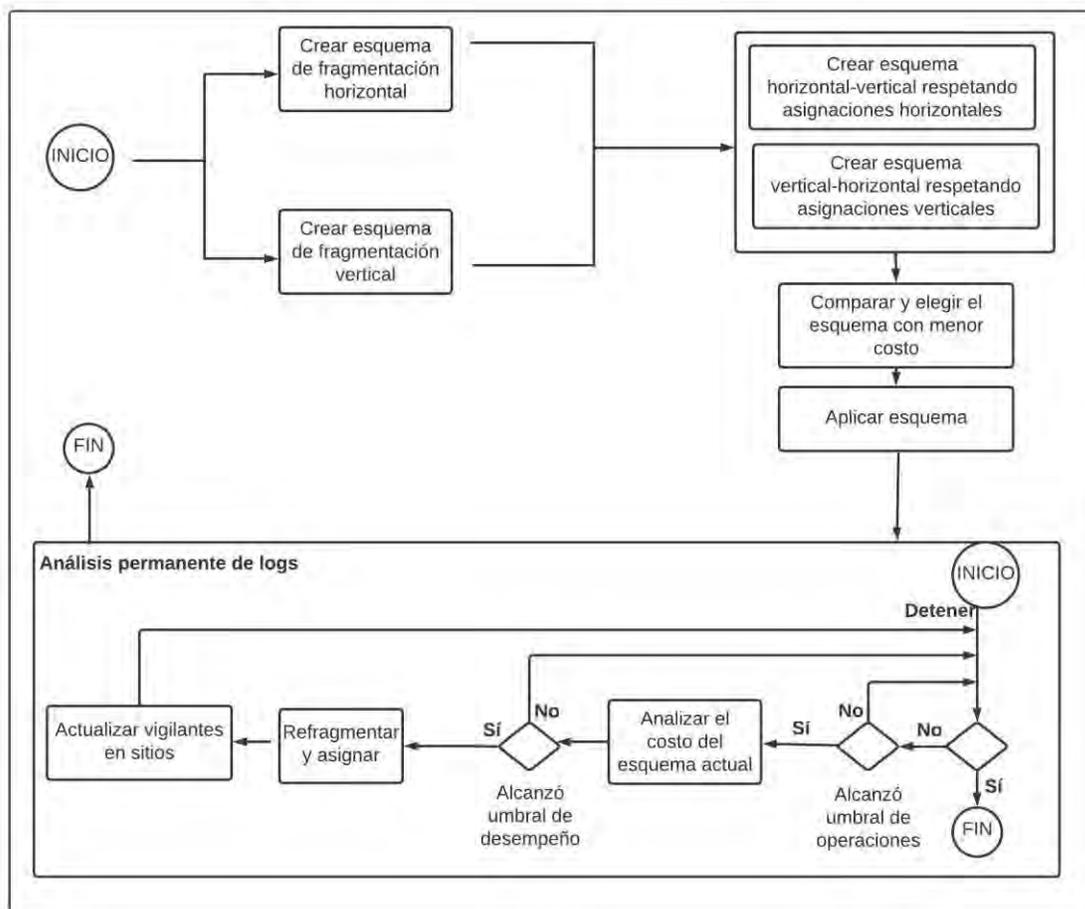


Figura 3.2 Flujo de trabajo de la fragmentación híbrida dinámica para bases de datos multimedia

En la Figura 3.3 se muestra el proceso que se lleva a cabo para realizar la fragmentación híbrida de forma más detallada. La unión de la fragmentación horizontal y vertical da como resultado la fragmentación híbrida.

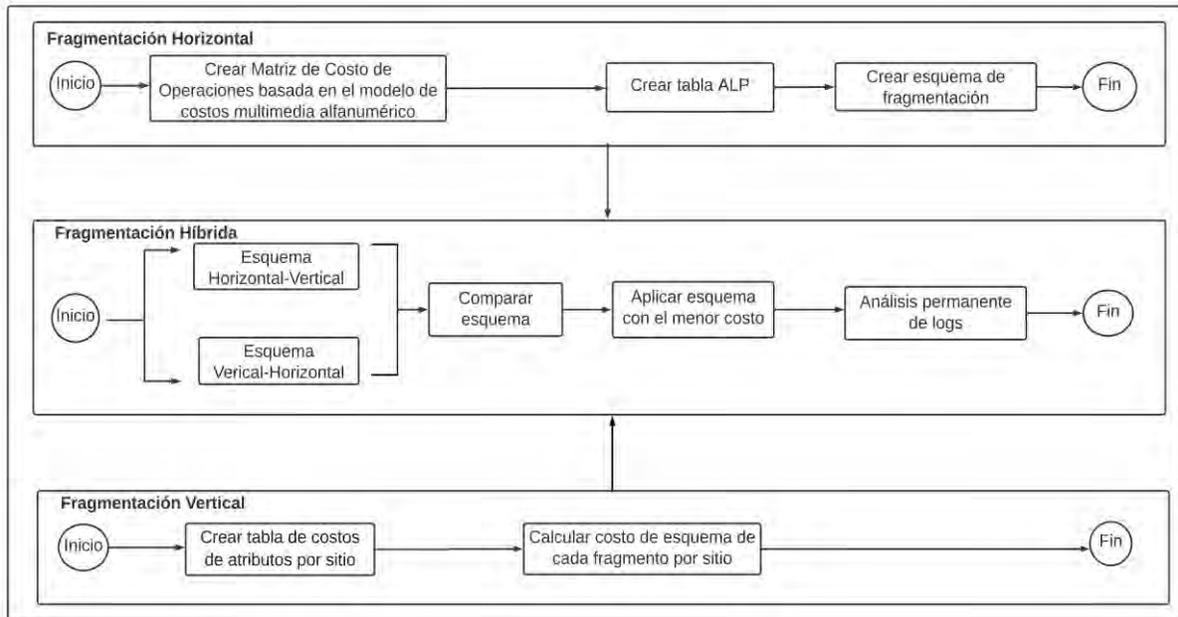


Figura 3.3 Flujo de proceso de fragmentación híbrida dinámica para bases de datos multimedia

El flujo del proceso de la fragmentación híbrida de la Figura 3.3 está conformado de la siguiente forma:

Los dos tipos de fragmentación híbrida son la fragmentación horizontal-vertical y la vertical-horizontal; sin embargo, solo un tipo de enfoque es el que se aplica. Para llevar a cabo la selección, se comparan los dos tipos de fragmentación y se elige el esquema con menor costo, como se muestra en la Figura 3.2.

1. Primero se muestra el proceso de la fragmentación horizontal que se conforma de dos pasos, los cuales se mencionan y describen a continuación:

Creación de matriz de costo de operaciones: Es una matriz que muestra el costo de operaciones realizadas en un sitio determinado bajo un cierto predicado.

El costo de operaciones ( $co$ ) se determina por la suma de los costos horizontales de las operaciones realizadas como se observa en la ecuación (1):

$$co = \sum_{i=1}^N Hor \tag{1}$$

**Donde:**

$i$  = cada predicado diferente utilizado en la tabla o fragmento.

$N$  = total de predicados.

El costo horizontal de operaciones ( $Hor$ ) se obtiene multiplicando el número de tuplas relacionadas con el predicado ( $tMax$ ) por el valor asignado de operaciones ( $VAO$ ) por el valor remoto ( $VR$ ) como se observa en la ecuación (2).

$$Hor = tMax * VAO * VR \tag{2}$$

El valor asignado de operaciones ( $VAO$ ) se presenta en la Tabla 3.2.

Tabla 3.2 Valores asignados a tipos de operaciones

Tipo de operación	Valor
Creación	2
Eliminación	2
Actualización	3
Lectura	1

El valor remoto ( $VR$ ) toma el valor de 1 cuando la operación es local y 2 si proviene de un sitio distinto al que contiene el fragmento requerido por la operación.

**Creación de la Tabla ALP:** La tabla ALP contiene el costo de cada atributo, determinado por la sumatoria de los costos de los predicados que contienen ese atributo. El esquema horizontal se obtiene mediante los predicados del atributo con el mayor valor ALP.

Al término de estos pasos, se logra la creación del esquema de fragmentación horizontal.

2. En segundo lugar, se tiene la fragmentación vertical (parte inferior de la Figura 3.3). Se crea la tabla de costo de atributos, que contiene información sobre qué atributo, en qué sitio es más requerido mediante el cálculo que multiplica el peso del atributo ( $Ta$ ) por el tipo de operación ( $VAO$ ) y el costo de reunión ( $CR$ ) en el cual, cuando las operaciones usan atributos que no están presentes en la misma tabla, se obtiene el valor de  $n + 1$ , donde  $n$  es el número de reuniones, de acuerdo a la ecuación (3). Por lo tanto, el costo vertical de una operación es la sumatoria del costo de atributos ( $Ca$ ) de acuerdo al número de atributos ( $M$ ) utilizados en esa operación como se observa en la ecuación (4).

$$Ca = \sum_{u=1}^H VAO_u * CR_u * VR_u * Ta \quad (3)$$

$$Ver = \sum_{j=1}^M Ca_j \quad (4)$$

**Donde:**

$u$  = cada operación realizada.

$H$  = total de operaciones.

Posteriormente, se calcula el costo de cada fragmento sumando los costos por cada atributo y finalmente se crea el esquema de fragmentos por sitio.

3. Finalmente, en la parte de en medio del diagrama, se encuentra el proceso para la fragmentación híbrida. Al obtener los dos esquemas Horizontal-Vertical y Vertical-Horizontal, se elige el de menor costo y se aplica con base en la ecuación (5); donde el costo híbrido de una operación es igual al promedio de su costo horizontal y vertical.

$$Hib = \frac{(Hor + Ver)}{2} \quad (5)$$

La parte dinámica de la fragmentación híbrida inicia con el análisis permanente de *logs* como se describió en la Figura 3.2.

La Figura 3.4 representa la arquitectura seleccionada la cual es MVC (Modelo Vista Controlador), en la vista se encuentran las páginas que interactúan con el usuario para solicitar información sobre las bases de datos y el acceso a ellas. Los *beans* administrados de JSF gestionan el flujo de datos que están en el controlador. En el modelo, se definen las reglas de negocio, que en este caso concreto será la aplicación de la fragmentación híbrida dinámica, además, de contemplar el cálculo de costos y la generación del vigilante-fragmentador para un gestor de contenido multimedia.

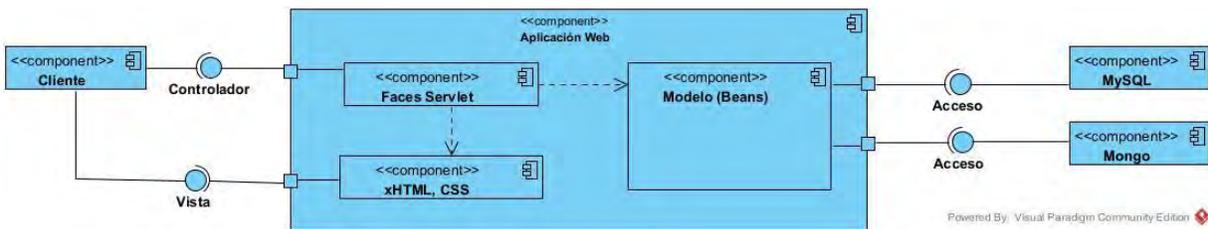


Figura 3.4 Arquitectura de la aplicación Web para realizar la fragmentación híbrida dinámica

### 3.3.1 Análisis de requisitos

Para establecer los requisitos funcionales de la aplicación Web, es necesario realizar el análisis de requisitos; estos requisitos se representan por medio de diagramas los cuales se muestran en esta sección.

El diagrama de casos de uso indica cómo se relacionan los usuarios con el sistema (actor-actividades); limitando las responsabilidades de cada usuario. En la Tabla 3.3 se definen los actores de la aplicación Web y la descripción del papel que desempeñan.

Tabla 3.3 Actores para la fragmentación híbrida dinámica

Actor	Actividad
<b>Administrador de la base de datos (DBA)</b>	El DBA ingresará la información y los archivos necesarios para la fragmentación.
<b>Vigilante-fragmentador</b>	El vigilante-fragmentador tendrá la función, a través de la información ingresada por el DBA, de realizar el proceso de manera automática, ejecutándose en el servidor en que se encuentre la base de datos a fragmentar.

Conforme a la definición y descripción de los actores de la Tabla 3.3, la Figura 3.5 muestra el diagrama de casos de uso de la fragmentación híbrida dinámica.



Figura 3.5 Diagrama de casos de uso para la aplicación Web y la fragmentación híbrida dinámica

El caso de uso “Fragmentar y asignar” representado en la Figura 3.5, se observa que tiene el estereotipo *Process* (Proceso) de acuerdo a lo establecido por la metodología UWE, por tal motivo, dicho proceso se representa por medio de un diagrama de actividades tal como se observa en la Figura 3.6.

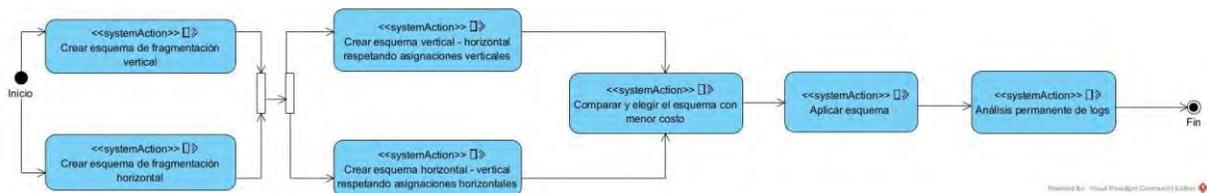


Figura 3.6 Diagrama de actividad del caso de uso “Fragmentar y asignar”

En el diagrama de la Figura 3.6 se crean los dos tipos de esquemas (vertical y horizontal), posteriormente, se combinan produciendo esquemas vertical-horizontal y horizontal-vertical, los cuales se comparan para elegir y aplicar el de menor costo. La parte dinámica se inicia con el análisis permanente de *logs*.

### 3.3.2 Diseño

A continuación, se presentan los modelos: conceptual, de navegación, presentación y proceso.

### 3.3.2.1 Modelo conceptual

El modelo conceptual, también conocido como modelo de dominio, describe la relación de los requisitos de la aplicación para obtener el comportamiento del sistema. En este trabajo, el modelo conceptual de la base de datos se representa por medio de tres diagramas: diagrama conceptual, diagrama lógico y diagrama físico, los cuales se presentan en las Figuras 3.7, 3.8 y 3.9.

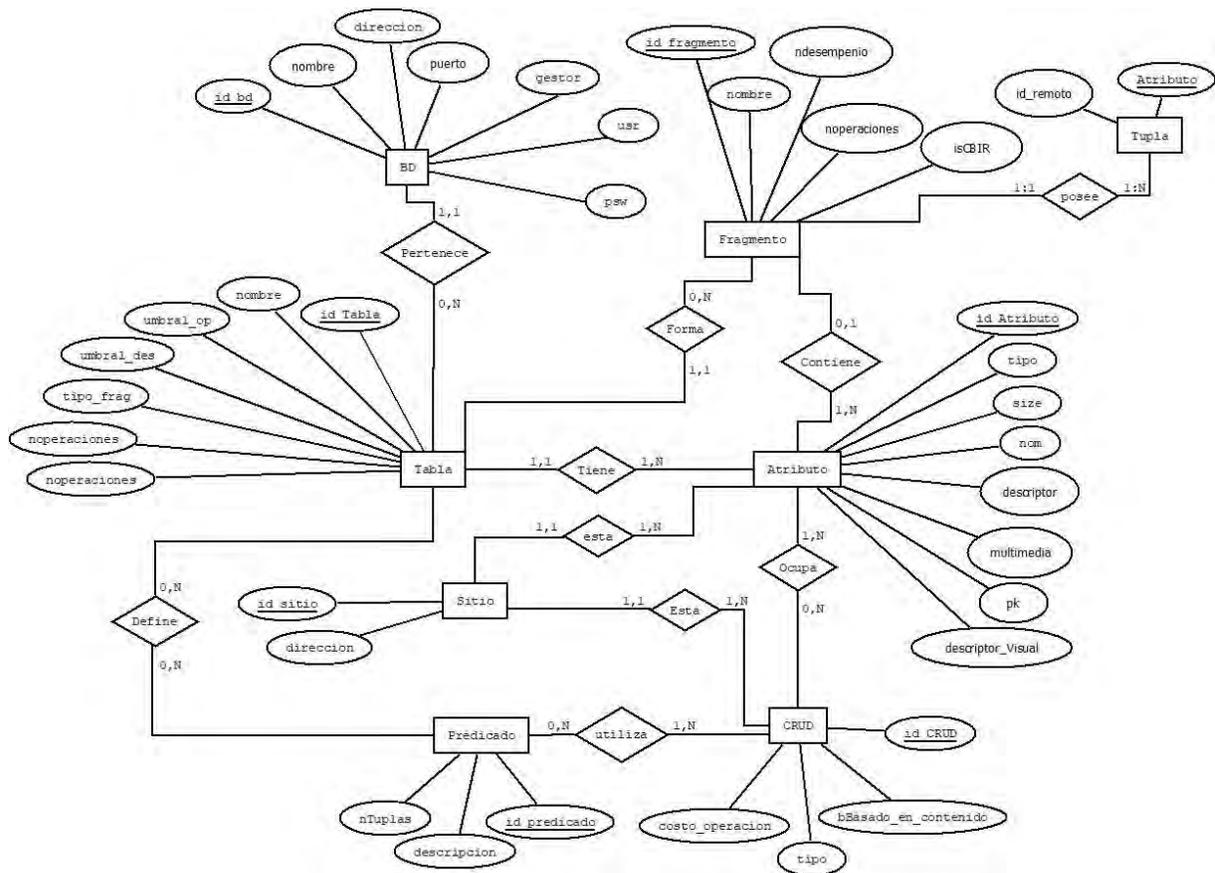


Figura 3.7 Diagrama conceptual de la aplicación

En el diagrama lógico (Figura 3.8) el atributo “descripción” de la entidad “Predicado” se muestra de tipo *varchar*, pero es de tipo *longtext* como se presenta en el diagrama físico de la Figura 3.9, esto es, porque la herramienta no permite agregarlo de tipo *longtext*.

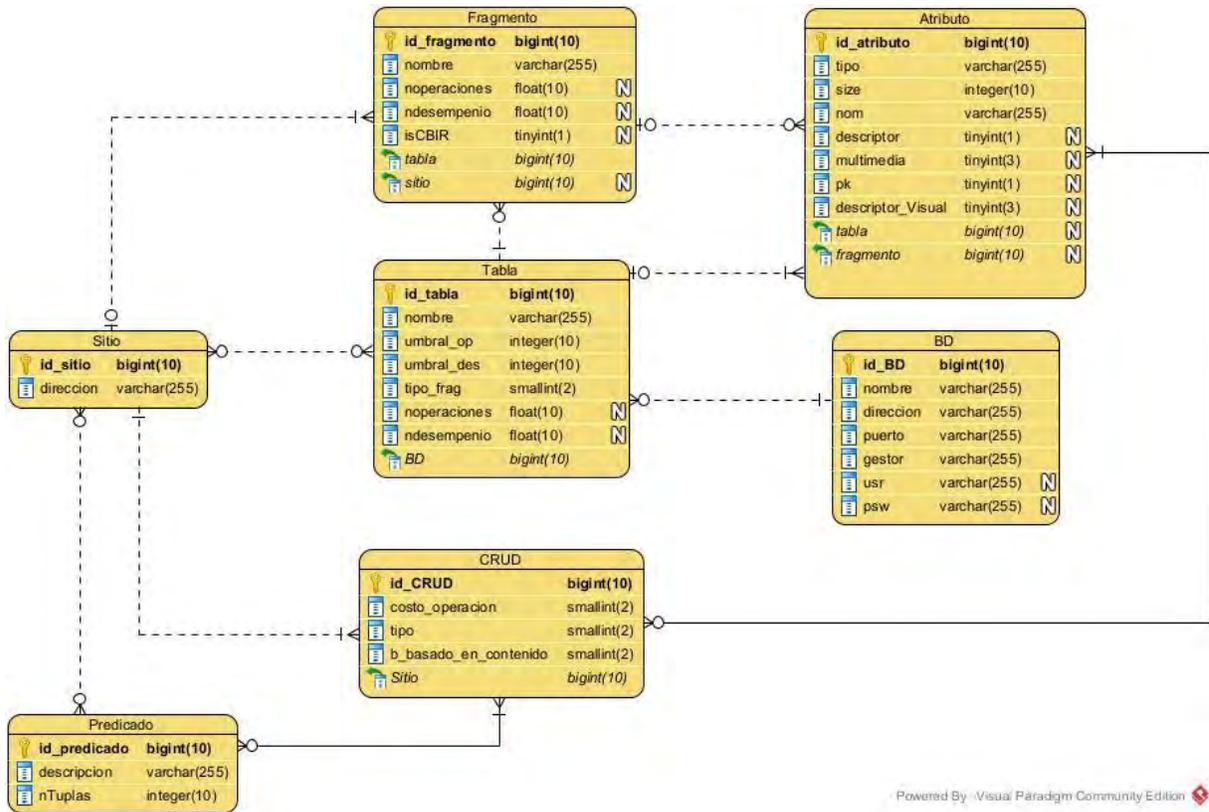


Figura 3.8 Diagrama lógico

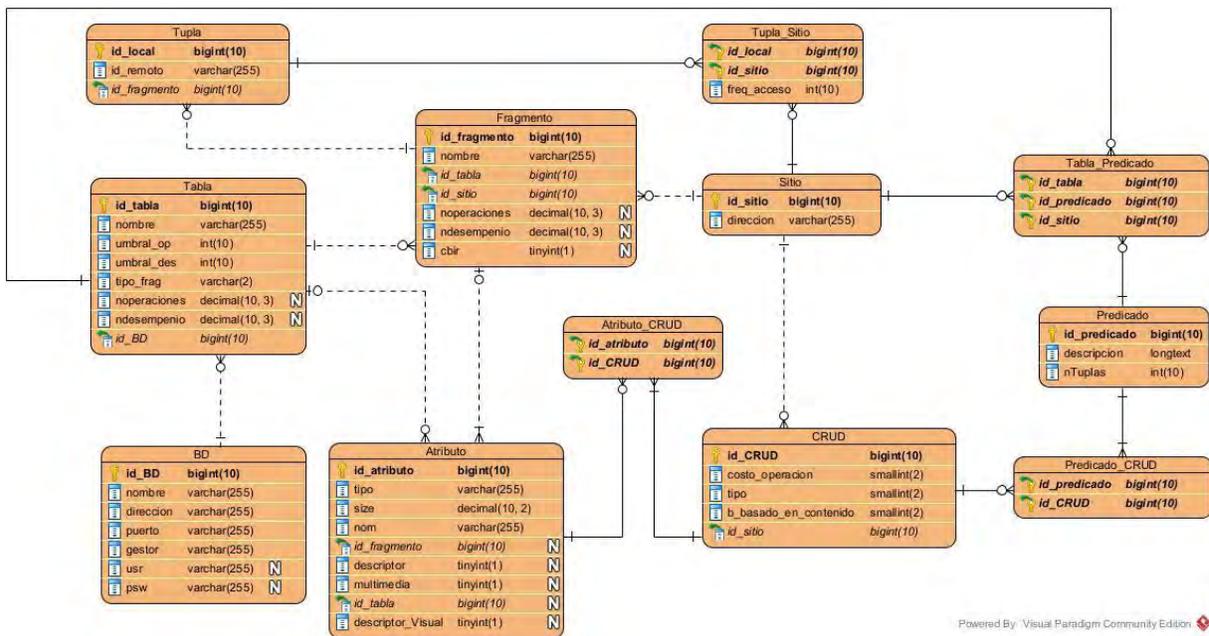


Figura 3.9 Diagrama físico

### Tablas de la base de datos

**Tabla:** La relación *Tabla* es la encargada de almacenar los datos de la tabla a fragmentar y re-fragmentar, como el nombre, el tipo de fragmentación que se realizará y los umbrales de operaciones y desempeño.

**BD:** La tabla *BD* almacena la información (nombre de la bd, dirección, puerto, gestor, usuario y contraseña) de la conexión hacia las bases de datos en la que se realizará la fragmentación híbrida dinámica. Se contempla que la contraseña sea opcional si el SGBD permite la deshabilitación de la autenticación.

**Atributo:** La tabla *Atributo* almacena el nombre, el tipo y el tamaño de los atributos de la tabla que se fragmenta.

**Fragmento:** La tabla *Fragmento* almacena información de los fragmentos resultantes de manera estática y dinámica al aplicar el método, de igual forma se almacena el número de operaciones que se considera en dicho fragmento. La tabla toma en cuenta los valores de operaciones y desempeño iniciales. Estos valores se utilizan para determinar el momento en el que los valores actuales superan los umbrales.

**Sitio:** La relación *Sitio* almacena la dirección IP de todos los sitios (remotos y locales) utilizados en el archivo *log* de cada SGBD.

**CRUD:** La tabla *CRUD* es la encargada de almacenar cualquier operación registrada en el archivo del historial de consultas de la base de datos que se relaciona con el predicado. Se registra en esta tabla el tipo de operación: lectura, creación, actualización y eliminación, el costo de la operación y si se considera CBIR.

**Predicado:** La tabla *Predicado* almacena todos los predicados de cada CRUD (*Create, Read, Update y Delete*, Creación, Lectura, Actualización y Eliminación) que se relaciona con la tabla a fragmentar y asignar. Los predicados son ampliamente ocupados en la fragmentación horizontal. Con esta tabla se construye la matriz MCRUD.

***Predicado\_CRUD:*** Esta tabla se encarga de mantener la cardinalidad entre las tablas *Predicado* y *CRUD*, ya que un CRUD posee la característica de tener muchos predicados y un predicado de estar en muchos CRUD.

***Atributo\_CRUD:*** La tabla *Atributo\_CRUD* relaciona todos los atributos utilizados en cada operación CRUD, de esta manera un atributo tiene la oportunidad de estar en muchas operaciones CRUD y una operación CRUD de utilizar muchos atributos.

***Tupla\_Sitio:*** Se encarga de almacenar la frecuencia de acceso que se tiene en la tupla por cada sitio y de igual forma mantener la cardinalidad entre las tablas.

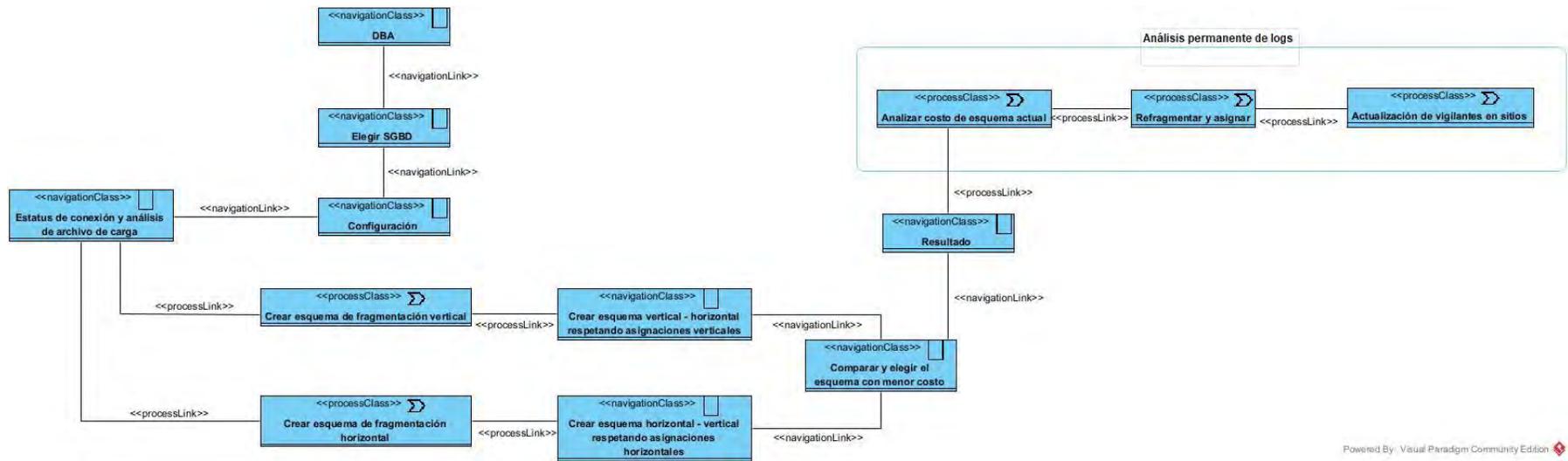
***Tupla:*** La tabla *Tupla* se encarga de almacenar información de todas las tuplas que se encuentran en un fragmento, así como un id remoto, un id local y el tamaño de la tupla en bytes.

***Tabla\_predicado:*** Relaciona el predicado de las tuplas con el identificador del sitio y tabla.

### 3.3.2.2 Modelo de navegación

Este modelo tiene como objetivo conocer la navegación de los objetos dentro de la aplicación, es decir, los caminos que el usuario utiliza para realizar una acción determinada y se representa por medio de clases. En la Figura 3.10 se presenta el modelo de navegación de la aplicación.

El administrador de la base de datos (DBA) elegirá el SGBD. “*Elegir SGBD*” es la encargada de permitir la elección del tipo de gestor (MySQL, Postgres-XL, PostgreSQL, y MongoDB) conectándose con la siguiente clase “*Configuración*”, donde el usuario ingresa los datos necesarios (dirección IP, puerto, nombre de la base de datos, usuario, contraseña y el nombre de la tabla a fragmentar) para hacer la conexión a la base de datos que se quiere fragmentar. Posteriormente, en la siguiente clase de navegación se encuentra el estatus de conexión, cuando la conexión sea exitosa, el paso siguiente es seleccionar el tipo de fragmentación (horizontal, vertical e híbrida), asignando los umbrales de operación y desempeño, así como la obtención del archivo de carga. Las siguientes clases de navegación corresponden a la fragmentación vertical y horizontal. Se crea el esquema vertical-horizontal respetando asignaciones verticales y el esquema horizontal-vertical respetando asignaciones horizontales.



Powered By: Visual Paradigm Community Edition

Figura 3.10 Modelo de navegación de la aplicación Web

Comparar y elegir el esquema con menor costo analiza ambas mezclas para realizar su objetivo. Una vez terminados los procesos anteriores, se presenta el esquema de fragmentación resultante de la tabla fragmentada, donde se tiene la opción de aplicar esquema. Al finalizar la aplicación del esquema de fragmentación, se entrega un *token*, este permite iniciar con el análisis permanente de *logs* (vigilante-fragmentador) para realizar la fragmentación dinámica.

El análisis permanente de *logs* se ejecuta de manera interna y se estructura de la siguiente manera: la clase “*Analizar costo del esquema actual*” es donde se realiza un análisis del costo del esquema actual con base en la carga de trabajo y los fragmentos, si el costo supera el umbral de desempeño se lleva a cabo el proceso de refragmentación y asignación, por último, se tiene el proceso de actualización de vigilantes en sitios.

### 3.3.2.3 Modelo de presentación

El modelo de presentación representa la interfaz de usuario de una aplicación Web de forma abstracta y se basa en el modelo de navegación. En la Figura 3.11 se modela la página de “*Configuración*” que contiene los siguientes elementos: *menu*, *button*, *form*, *fileUpload*, *textInput*, *slider* y *selection*. El objetivo de esta página es seleccionar el SGBD y la configuración necesaria para iniciar con el proceso de la fragmentación, a continuación se describe detalladamente.

“*Elegir SGBD*” es un menú con las opciones del SGBD (MySQL, PostgreSQL y MongoDB), donde el usuario selecciona el gestor que utilizará para llevar a cabo la fragmentación. En seguida se tiene un formulario con las opciones de: dirección IP, puerto de la base de datos, usuario, contraseña y tabla a fragmentar; para esto, se tiene que realizar el proceso de “*Estatus de conexión*” el cual se representa en la Figura 3.12. Si la conexión fue exitosa, se selecciona el tipo de fragmentación, archivo *log*, umbral de operaciones y umbral de desempeño.

La Figura 3.12 se encuentra conformada por dos botones (*Probar conexión* y *Cancelar*). De acuerdo al estado se le notifica al usuario por medio de un mensaje. Si la conexión es exitosa, se procede con los detalles de la fragmentación, Figura 3.13.

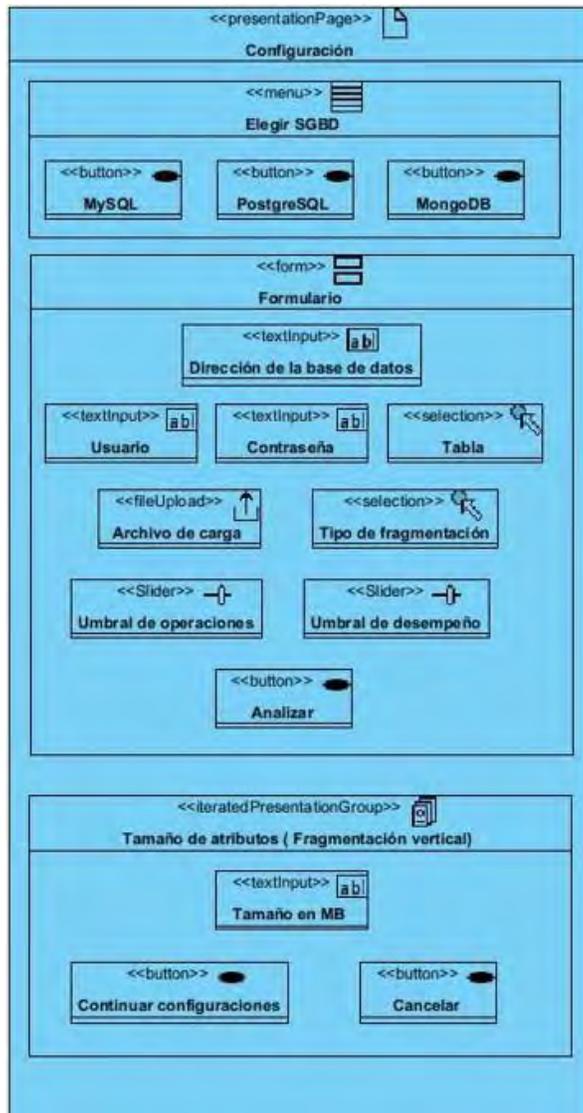


Figura 3.11 Página *Configuración* de conexión y fragmentación del modelo de presentación

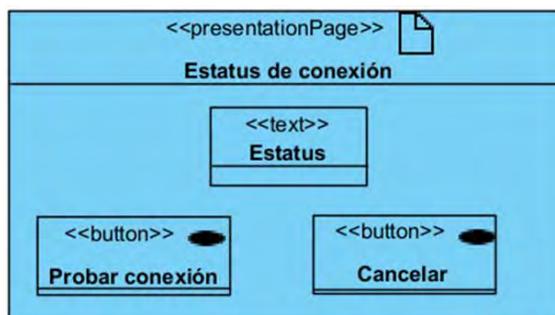


Figura 3.12 Página *Estatus de conexión*



Figura 3.13 Página *Detalles de la fragmentación híbrida*

El modelo de presentación “*Detalles de la fragmentación híbrida*” se encuentra estructurado por tres “*presentationAlternatives*”. En el primero, se detalla el proceso de la fragmentación horizontal iniciando por el recuadro *Selección de esquema*, este solo se aplica en caso de utilizar CBIR. Se crea la MCRUD de acuerdo al costo de operaciones realizadas en un sitio bajo el predicado establecido. Después, se crea la tabla ALP la cual contiene el costo de cada atributo con base en la sumatoria de los costos de los predicados. Finalmente, se presentan los resultados finales para la fragmentación horizontal.

En el segundo “*presentationAlternatives*”, se tiene los detalles de la fragmentación vertical, se crea la tabla de costo de atributos, que contiene información sobre qué atributo, en qué sitio es más requerido, se calcula el costo de cada fragmento sumando los costos por cada atributo, se crea el esquema de fragmentos por sitio y se genera la tabla final.

En el tercer “*presentationAlternatives*” (*Fragmentación híbrida*), se obtienen los dos esquemas vertical-horizontal y horizontal-vertical, después se elige el esquema de menor costo.

Además, se tienen dos botones (*Regresar, Fragmentar y Asignar*), donde “*Fragmentar y Asignar*” aplica el esquema de fragmentación, se presenta un diálogo con el estatus de la acción realizada, mostrando una vista resumida de los fragmentos finales. La página relacionada con esta actividad se presenta en la Figura 3.14.



Figura 3.14 Página de resultados del modelo de presentación

### 3.3.2.4 Modelo de proceso

El modelo de proceso permite describir el comportamiento de una clase de proceso; en este diagrama se representa la acción del sistema por medio del estereotipo “*systemAction*” y la acción del usuario por “*userAction*”. Además, este modelo se basa en el diagrama de actividades. En la Figura 3.15 se observa el modelo de proceso de la fragmentación híbrida en el cual se representan las clases *BD* y *Tabla*. Además, se muestra el recuadro del análisis permanente de los *logs*.

La primera acción del sistema, se inicia con un formulario para capturar toda la información requerida tal como: SGBD, dirección de la base de datos, usuario, contraseña, archivo *log*, tabla, umbral de operaciones, umbral de desempeño y el tipo de fragmentación (horizontal, vertical e híbrida).

Si el sistema gestor seleccionado es MongoDB y el tipo de fragmentación es vertical o híbrida, el usuario debe asignar el tamaño de atributos.

El usuario ingresa los datos y realiza la prueba de conexión; si los datos son correctos, el segundo proceso es cuando la aplicación crea el esquema de fragmentación vertical y horizontal para, posteriormente, llevar a cabo la fragmentación híbrida, en donde la aplicación crea el esquema vertical-horizontal respetando asignaciones verticales y horizontal-vertical respetando asignaciones horizontales. Se compara y elige el esquema con menor costo y se aplica el esquema; estas funciones las realiza XAMANA, sin embargo, solo produce esquemas estáticos.

La parte dinámica inicia con el análisis permanente de *logs* y se llevan a cabo las siguientes actividades por parte del sistema: ¿Alcanzó el umbral de operaciones? Si no lo alcanzó, se termina el proceso, caso contrario, la acción es analizar el costo del esquema actual, ¿Alcanzó el umbral de desempeño? Si es afirmativo, se refragmenta y asigna, actualizando vigilantes en sitios. Por lo tanto, la fragmentación híbrida utiliza y aplica el esquema combinado más barato bajo la carga de trabajo encontrada en el *log*.

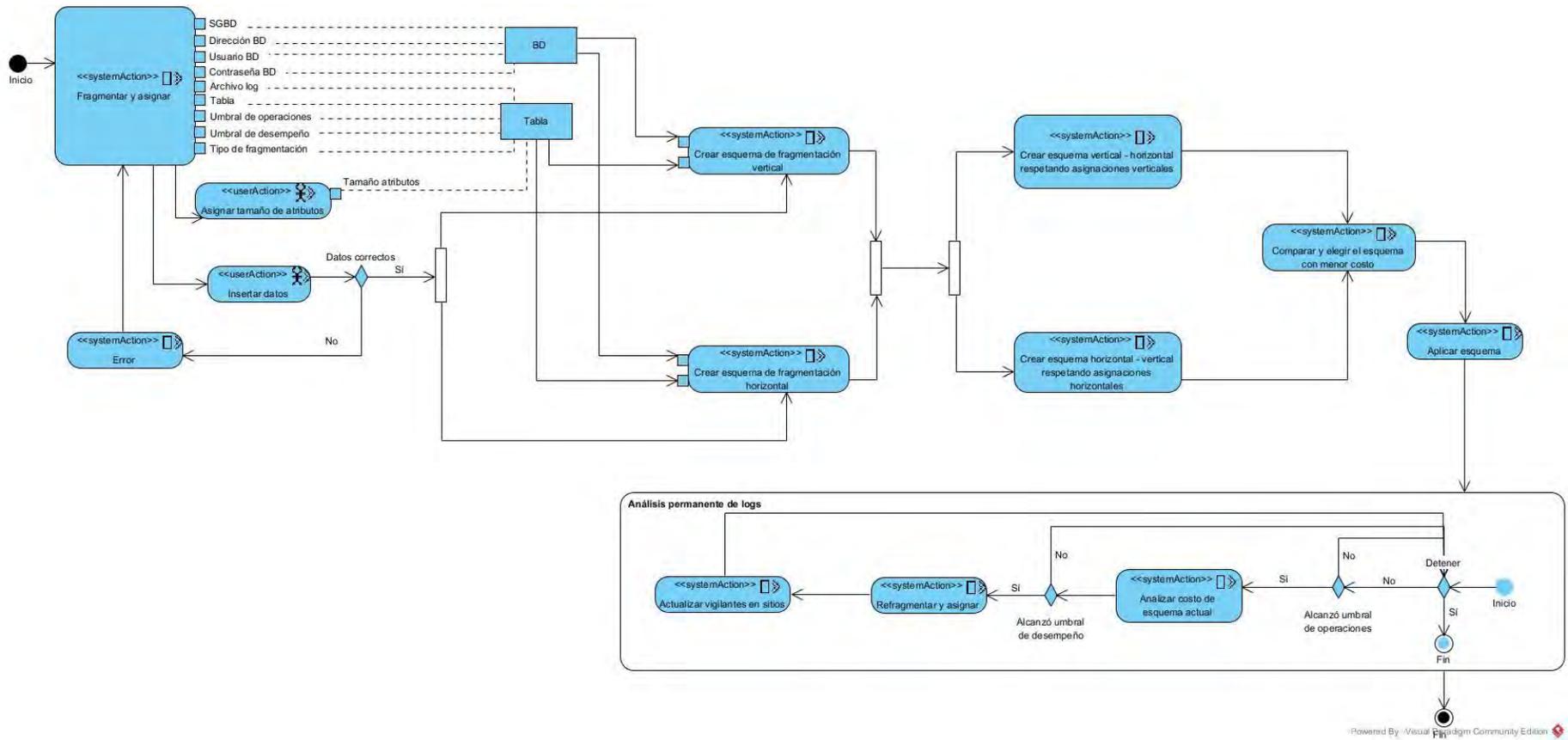


Figura 3.15 Diagrama de proceso de la fragmentación híbrida

## Capítulo 4. Resultados

En este capítulo se presentan los resultados obtenidos del análisis del estado del arte, el funcionamiento de la aplicación Web que permite obtener la fragmentación estática, para posteriormente, mostrar la implementación del método de fragmentación híbrida dinámica, a través del vigilante-fragmentador. Por último, la comparación de la técnica propuesta con dos métodos seleccionados.

### 4.1 Resultados del análisis

Para finalizar la etapa del análisis, en esta sección se muestran detalladamente los resultados obtenidos de los trabajos seleccionados en la sección 3.1, mediante gráficas, clasificándolos por 11 cualidades.

La Figura 4.1 muestra el número de artículos por año de publicación. El mayor número de artículos (19) se encuentra entre los años 2018 a 2022, donde el 32% usó un *benchmark*, solo el 47% están completos y 42% son fáciles de implementar, mientras que, 37% se basan en un modelo de costos, en el 21% la fragmentación es dinámica y 16% contemplaron datos multimedia.

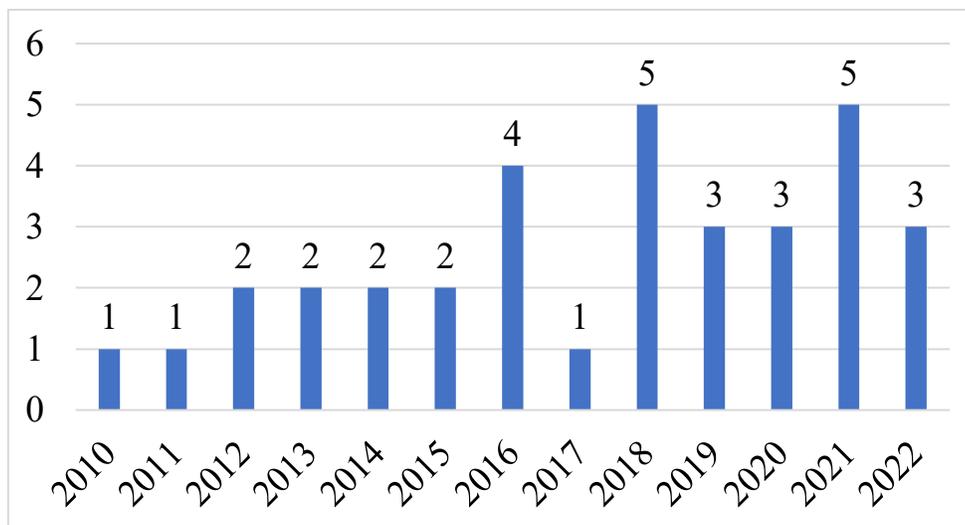


Figura 4.1 Número de artículos por año de publicación

La Figura 4.2 muestra una comparación de los artículos por editorial, donde la mayoría de los trabajos fueron publicados por Springer en comparación con ACM, IEEE y Otras. En el caso de Springer, solo Rani et al. [71] consideraron ambos esquemas de fragmentación (estática y dinámica), mientras que la característica principal que se destaca en Kling et al. [58], Jindal & Dittrich [59], Gorla et al. [60], Song & Chen [61] y Patel et al. [84] es el uso de un modelo de costos. Los artículos clasificados en “Otros”, Chbeir et al. [57] y Rodríguez-Mazahua et al. [68], utilizaron datos multimedia, por otra parte, Chen et al. [65] y Wang et al. [63] consideraron una fragmentación dinámica, mientras que, Koong et al. [74], Azila et al. [83] y Cantini et al. [86] solo reúnen un solo criterio.

Por otro lado, con respecto a los artículos publicados en IEEE, Awad et al. [36], Badran et al. [78] y Chen et al. [62] no cumplen con las características de completitud, facilidad de implementación y modelo de costos, es decir, tienen de cero a dos criterios, y solo Noraziah et al. [23] y Vogt et al. [76] son dinámicos.

Finalmente, en ACM se aplicaron bases de datos diferentes a las tradicionales sin tener en cuenta datos multimedia (Durand et al.[73] y Padiya et al. [69]); por ejemplo: en Schreiner et al. [16] el tipo de base de datos utilizada fue NewSQL

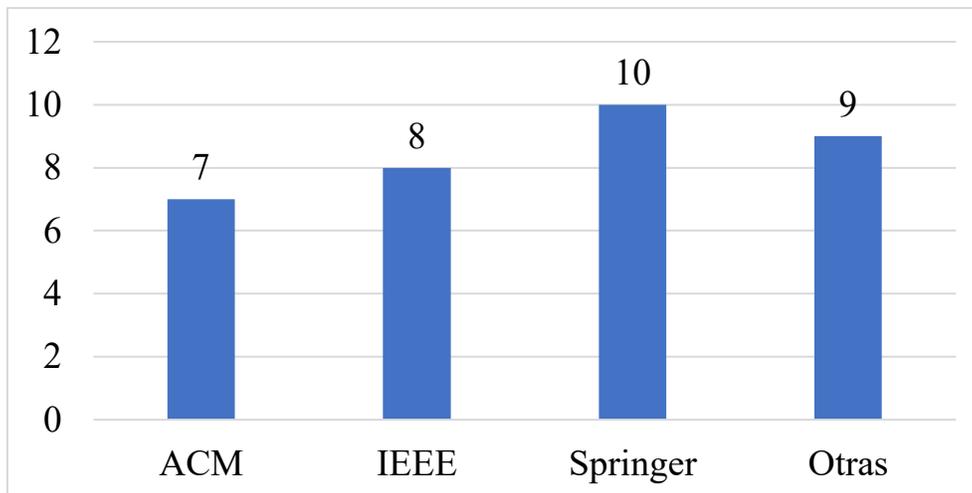


Figura 4.2 Número de artículos por editorial

El uso de *benchmarks* se presenta en la Figura 4.3, donde 16 artículos los aplicaron, como es, TPC-C y TCP-H, [59], [63], [81] . El resto de los trabajos usaron bases de datos para validar sus métodos. Por consiguiente, la Figura 4.4 compara el número de artículos de acuerdo al tipo de carga de trabajo, donde se encontró que, los más usados son TPC-C y TPC-H y el tipo de fragmentación fue dinámica. Mientras que, 18 trabajos no usaron algún tipo de *benchmark*, además, en la gráfica de la Figura 4.4 se observa que, algunos autores utilizaron dos cargas de trabajo, por ejemplo, Jindal & Dittrich [81] se enfocaron en TPC-H y SSB.

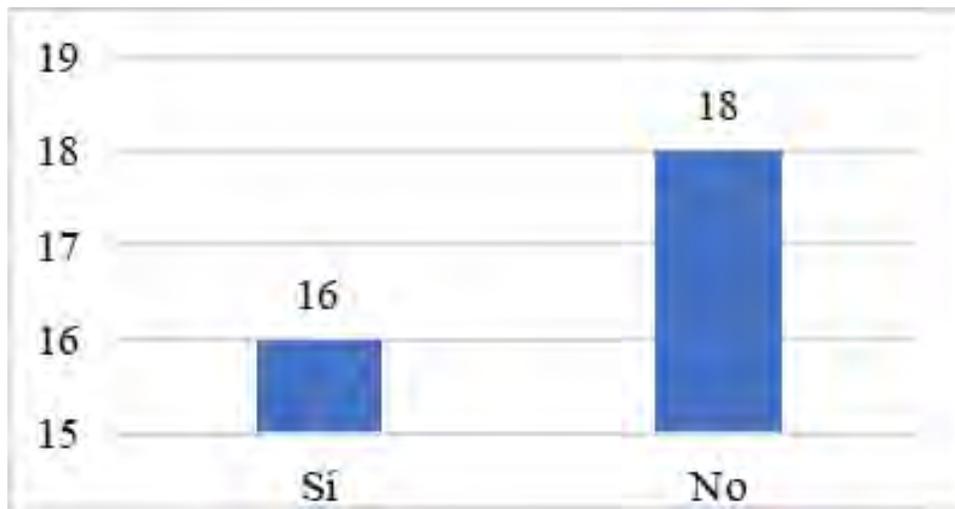


Figura 4.3 Número de artículos que usaron un *benchmark*

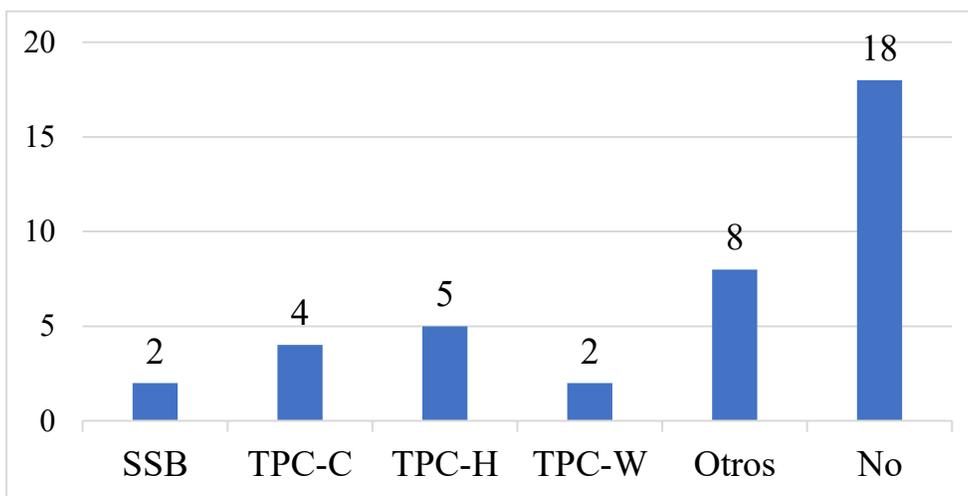


Figura 4.4 Número de artículos por tipo de *benchmark*

Para considerar un trabajo completo y fácil de implementar, se establecieron los siguientes dos criterios: 1) el artículo presenta el algoritmo, tecnologías y los resultados obtenidos al momento de implementar su propuesta y 2) si no requiere conocimientos especializados para su replicación. Las gráficas de las Figuras 4.5 y 4.6 muestran el resultado basado en esta clasificación, donde en la primera se observa que 22 trabajos de 34 son completos porque muestran todo el proceso y la información requerida para su replicación, mientras que en la segunda gráfica, 14 de 34 son fáciles de implementar porque no se requiere de algún conocimiento especializado, como *Machine Learning Technics* (Técnicas de Aprendizaje Automático).

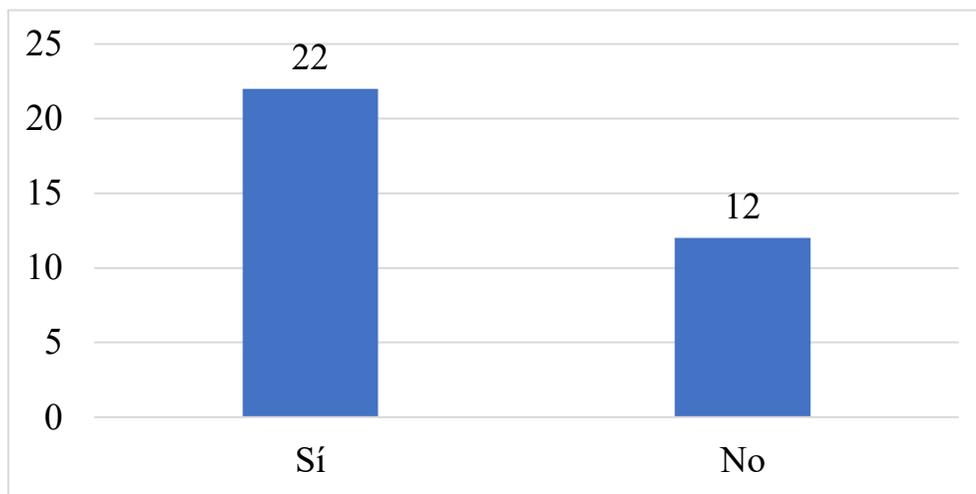


Figura 4.5 Número de artículos por completitud

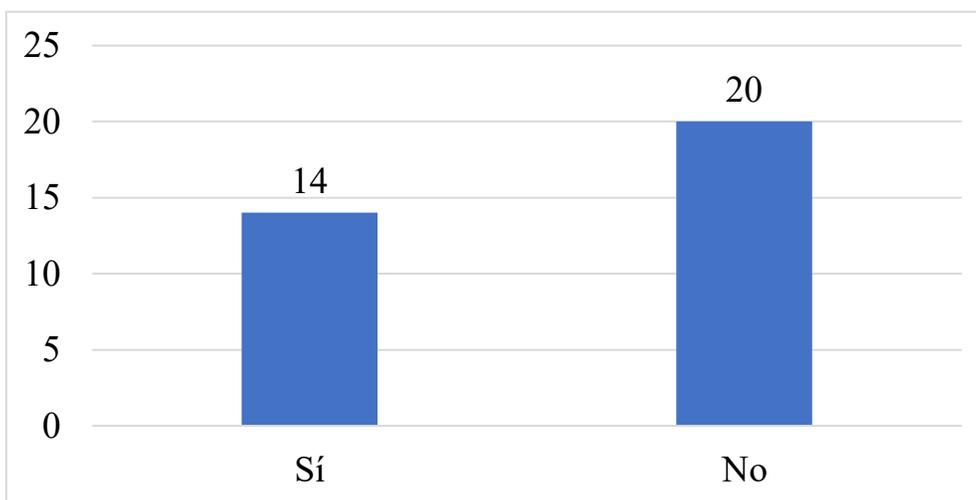


Figura 4.6 Número de artículos por facilidad de implementación

Los trabajos que consideraron un modelo de costos, se enfocaron en estimar el tiempo de procesamiento de las consultas, el acceso a disco y la comunicación entre los nodos, con la finalidad de evaluar el rendimiento del sistema al momento de fragmentar. Por ejemplo, Rodríguez Mazahua et al. [68] usaron el modelo de costos para evaluar el esquema de partición híbrida en bases de datos multimedia. La Figura 4.7 muestra los trabajos que implementaron un modelo de costos en su método, donde solo 18 usaron esta cualidad, por ejemplo, [76] y [84].

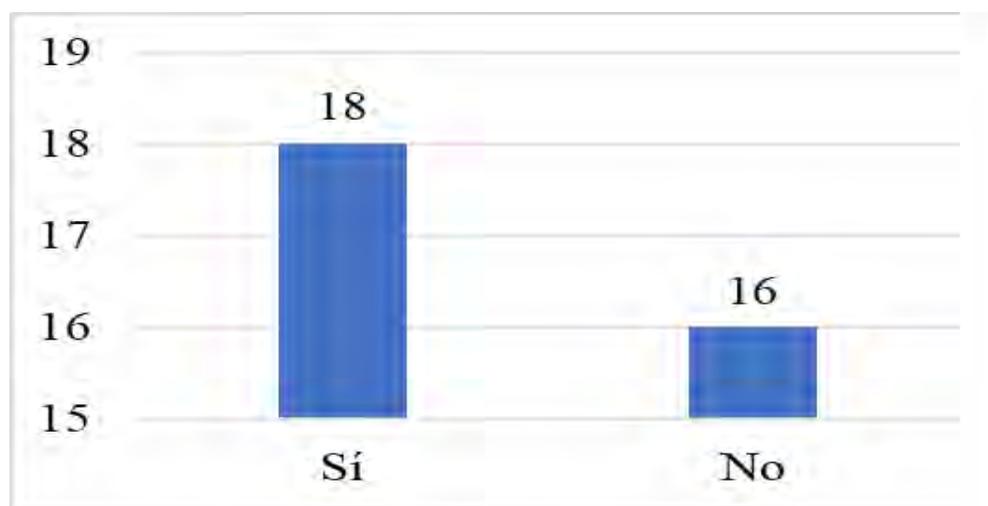


Figura 4.7 Número de artículos por modelo de costos

La gráfica de la Figura 4.8 muestra la comparación del número de artículos y el tipo de técnica (estática o dinámica) usada en los trabajos. Las técnicas que son capaces de adaptar el esquema de fragmentación híbrida a la carga de trabajo corresponden a métodos dinámicos y estos son: Noraziah et al. [23], Jindal & Dittrich [59], Wang et al. [63], Chen et al. [65], Rani et al. [71], Vogt et al. [76], Kulba & Somov [80] y Kang et al. [81]. Cuando la fragmentación no considera cambios en los patrones de acceso se considera como estática, como en [5], [57], [61], [68], [72] y [82]. Rani et al. [71] destaca porque en este artículo también se aplicó la parte estática y dinámica en la fragmentación híbrida.

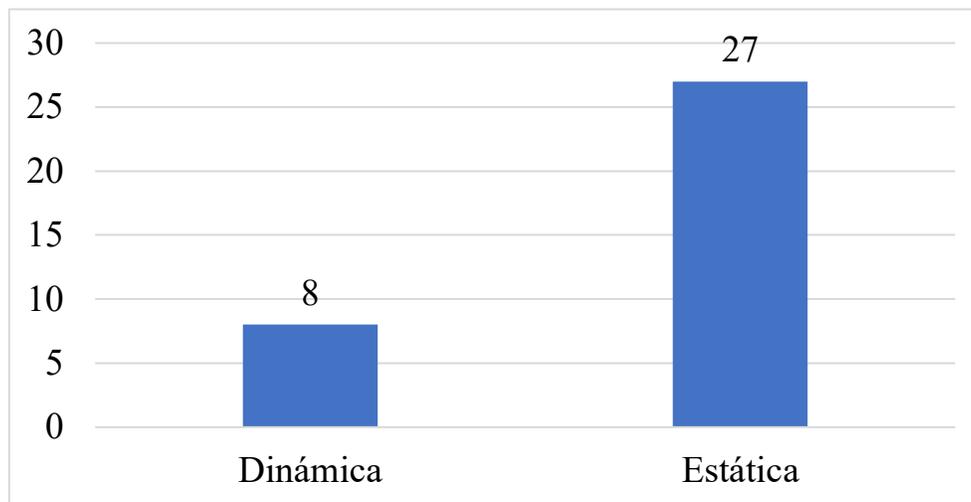


Figura 4.8 Número de artículos por tipo de fragmentación dinámica o estática

El tipo de base de datos usada en cada trabajo se muestra en la gráfica de la Figura 4.9. Como se observa, las bases de datos más utilizadas fueron relacionales, solo seis de estos 17 trabajos consideran bases de datos relacionales y fragmentación dinámica ([23], [59], [63], [65], [71], [81]). En contraste, dos reunieron cuatro criterios ([59] y [63]), pero no usaron datos multimedia. Mientras que, los trabajos que no se enfocaron en el uso de bases de datos relacionales solo consideran la fragmentación estática, cuatro toman en cuenta modelos de costos ([60]–[62] y [69]). Los trabajos de Safaei [5], Awad et al. [36] y Badran et al. [78] reúnen de cero a tres criterios.

Como se mencionó en el Capítulo 3, sección 3.1 Análisis, el proceso de análisis resultó en 34 artículos, sin embargo, en la Figura 4.9 se observa que, de acuerdo al tipo de repositorio, la suma da como resultado 44 trabajos, porque en algunos métodos se aplicaron más de un tipo de base de datos, por ejemplo, bases de datos relacionales y distribuidas.

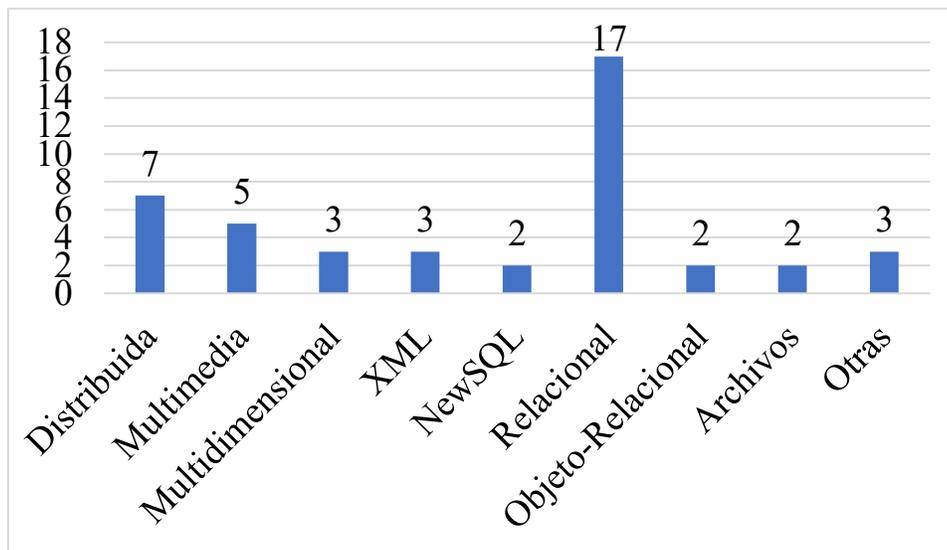


Figura 4.9 Número de artículos por tipo de repositorio

Las consultas basadas en contenido es otro criterio considerado dentro del análisis, sin embargo, la mayoría de los artículos no consideran este aspecto, y se observa en la Figura 4.10. Solo dos hacen uso de las consultas basadas en contenido: Chbeir et al. [57] y Mourão & Magalhães [72]. Ambos trabajos, aunque contemplaron datos multimedia, no cumplen con los criterios establecidos.

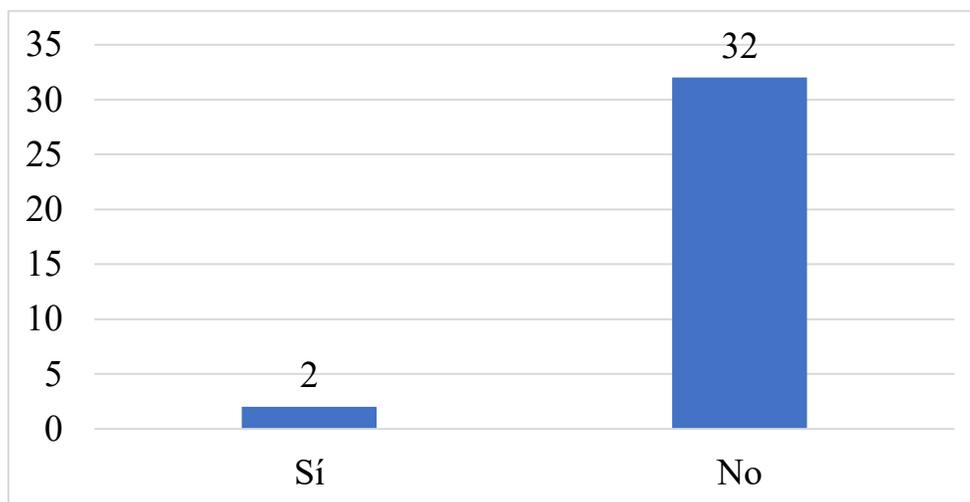


Figura 4.10 Número de artículos por consultas basadas en contenido

## 4.2 Base de datos HITO

HITO es una base de datos multimedia que se encuentra en MongoDB, la cual almacena la historia del Instituto Tecnológico de Orizaba, incluyendo imágenes y videos de los diferentes eventos, alumnos y personal docente, por mencionar algunos. Actualmente, se encuentra conformada de 19 colecciones. En la Figura 4.11 y 4.12 se muestran las colecciones de HITO.

Collection Name	Storage size	Documents	Avg. document size	Indexes	Total index size
alumnos	32.77 kB	406	122.00 B	1	20.48 kB
aportaciones	4.10 kB	0	0 B	1	4.10 kB
CBIR	4.10 kB	0	0 B	1	4.10 kB
con_buildings	100.66 MB	167	607.34 kB	2	45.06 kB
con_equipment	41.62 MB	104	401.69 kB	2	69.63 kB
con_events	255.99 MB	723	352.30 kB	2	81.92 kB
con_personal	449.32 MB	874	512.42 kB	2	98.30 kB
departamento	20.48 kB	39	132.00 B	1	20.48 kB
edificios	28.67 kB	20	1.05 kB	1	20.48 kB
eventos	53.25 kB	65	1.05 kB	1	20.48 kB

Figura 4.11 Colecciones de la base de datos HITO, parte 1

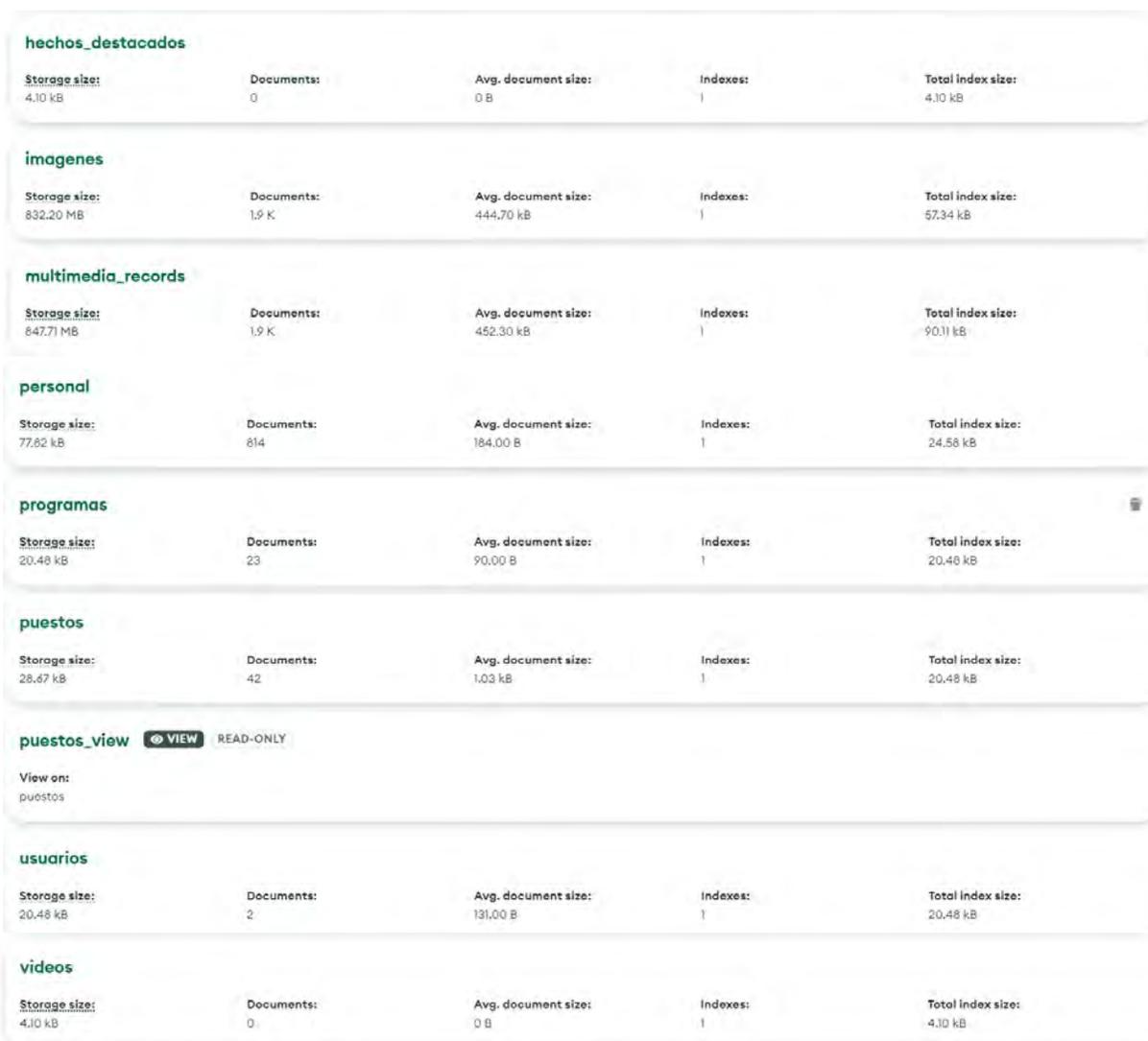


Figura 4.12. Colecciones de la base de datos HITO, parte 2

En este proyecto, se utilizó *multimedia\_records*, que es una colección de 1870 documentos, que almacena principalmente las imágenes de eventos, edificios, equipos o personal del Instituto Tecnológico de Orizaba. En la Figura 4.13 se muestra un documento de *multimedia\_records* como ejemplo. Se observa que cada documento tiene un identificador único, el nombre de la imagen, la descripción, la imagen, si se encuentra validado, el descriptor *SURF* de la imagen y el tipo.

```

_id: "IMG830"
nombre: "Laboratorio 10"
imagen: "iVBORw0KGgoAAAANSUHEUgAAALAAAAGQCAIAAADugMk7AAAACXBIXMAAASTAAAEwEAmp..."
descripcion: "Laboratorio de Quimica "
validado: "si"
▶ descriptor: Array (43392)
tipo: "building"

```

Figura 4.13 Un documento de la colección *multimedia\_records*

### 4.3 Aplicación Web XAMANA

En esta sección, se describe el funcionamiento de la aplicación Web XAMANA, la cual, permitirá llevar a cabo la fragmentación híbrida estática, para posteriormente, aplicar la fragmentación híbrida dinámica del método propuesto. En ambas fragmentaciones híbridas (estática y dinámica) se consideraron dos sitios y se asignaron a las cargas de trabajo:

- **Sitio 1:** 192.168.1.64 (donde se encuentra la base de datos HITO con las 19 colecciones)
- **Sitio 2:** 192.168.1.75

En un inicio la colección *multimedia\_records* se encuentra en el sitio 1, posteriormente, los fragmentos de esta se asignarán a los sitios a donde representen un mayor costo híbrido.

La Figura 4.14 muestra la vista inicial de XAMANA, que se conforma de una lista desplegable con las opciones de los cuatro SGBD, los cuales son: a) MySQL, b) Postgres-XL, c) PostgreSQL y d) MongoDB. El DBA selecciona el gestor de base de datos a utilizar y hace clic en el botón *Connection*; para este caso, se utilizará MongoDB.

Al hacer clic en *Connection*, se muestra el diálogo de configuración, donde se aprecia un formulario que permite capturar la información de la base de datos (dirección IP, puerto y nombre de la base de datos), información del administrador de la base de datos (usuario y contraseña) y una lista desplegable para seleccionar la relación o colección a fragmentar. MongoDB permite que la autenticación en sus bases de datos se deshabilite. Si la autenticación se encuentra deshabilitada, los campos relacionados (usuario y contraseña) son opcionales. Además, contiene dos botones, uno para comprobar la conexión (*Test connection*), y otro para cancelar (*Cancel*) el proceso, en la Figura 4.15 se presenta el formulario descrito anteriormente.



Figura 4.14 Página principal de la aplicación Web XAMANA

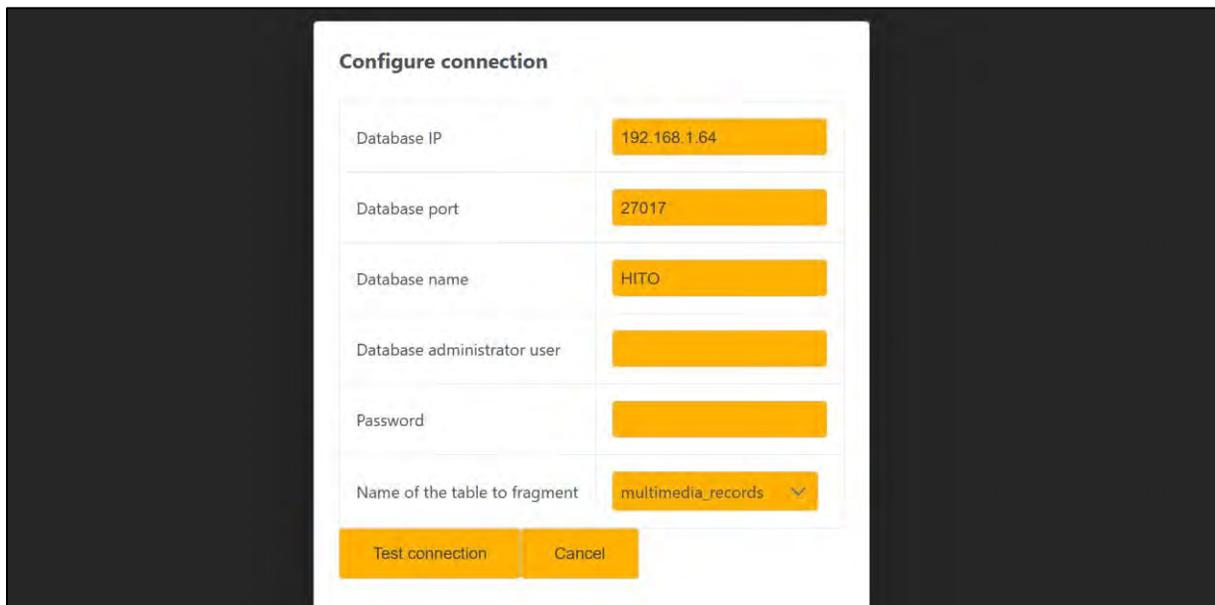


Figura 4.15 Página de configuración

Si los datos ingresados son correctos, se presenta un diálogo (Figura 4.16) indicando que la conexión a la base de datos fue exitosa y en la parte inferior se muestra un botón (*Configure fragmentation*, Configuración de fragmentación) que va a la siguiente página.

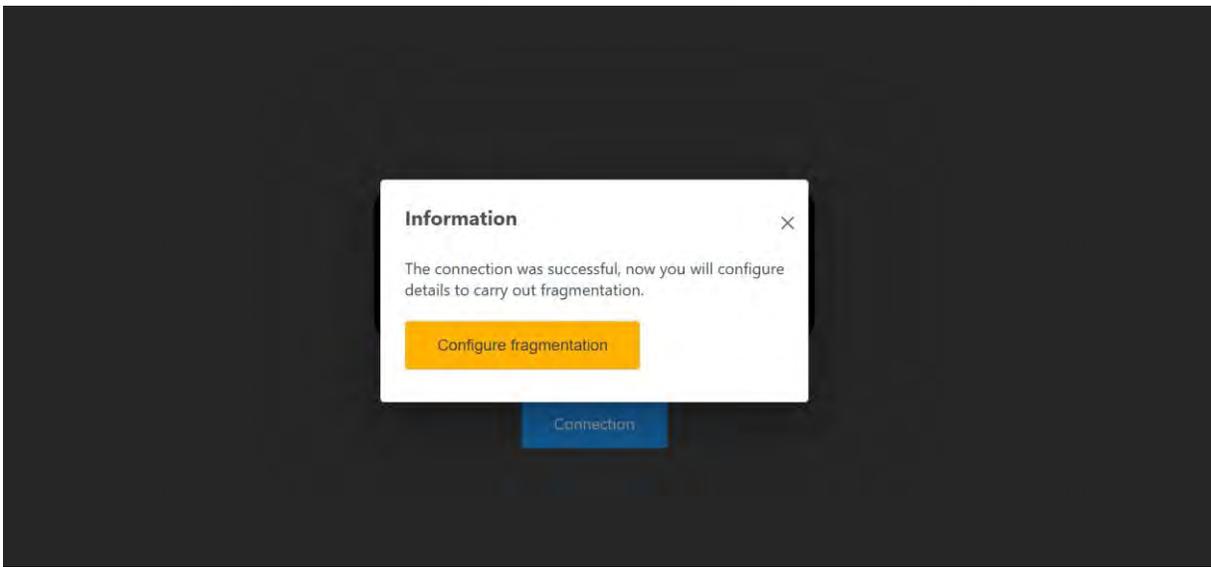


Figura 4.16 Modal de configuración exitosa

La Figura 4.17 muestra la vista “*Configuration, Configuración*”, esta permite asignar los valores de la fragmentación a realizar. En dicha pantalla, se muestra un formulario con las siguientes opciones: 1) lista desplegable para seleccionar cualquiera de los tres tipos de fragmentación (horizontal, vertical e híbrida), 2) si considera consultas basadas en contenido, 3) *choose* para cargar el archivo *log* con las operaciones de la base de datos, 4) *slider* para el umbral de operaciones y 5) *slider* para el umbral de desempeño.

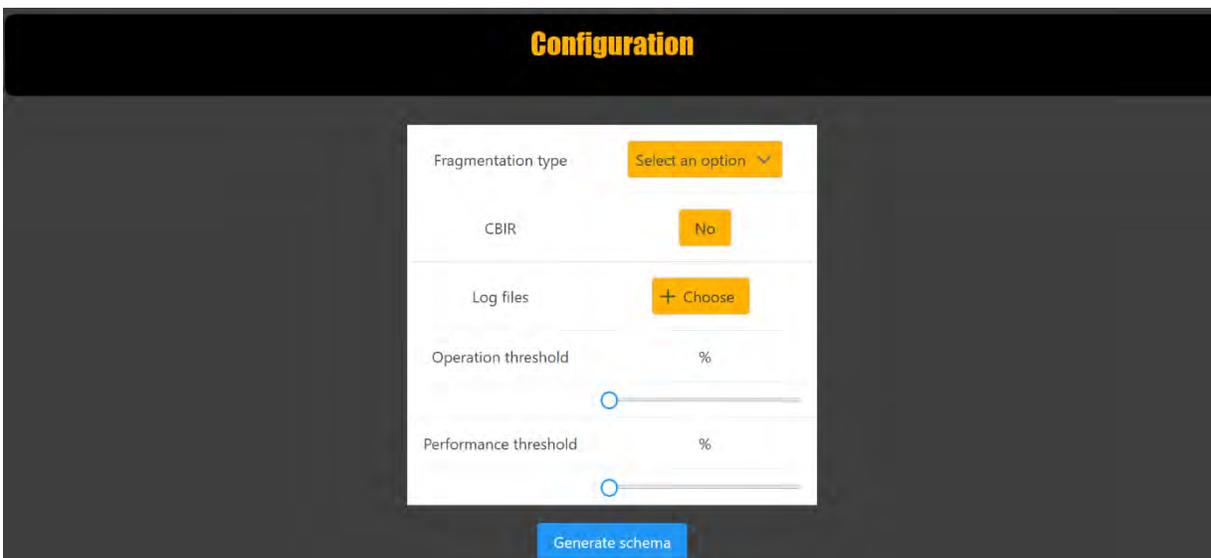


Figura 4.17 Página de configuración para la fragmentación

Como ya se mencionó, este trabajo se enfoca en la fragmentación híbrida, por tal motivo, la fragmentación a elegir es híbrida. MongoDB no define tamaños para cada tipo de dato, únicamente define el tamaño máximo de cada documento (16 MB). El modelo de costos utilizado en la fragmentación híbrida incluye el tamaño de los atributos en una de sus variables. El DBA debe asignar un tamaño máximo aproximado para cada campo de la colección.

Para asignar el tamaño de atributos se utiliza un modal donde se presenta una tabla con tres columnas, la primera indica el nombre del atributo, la segunda el tipo de dato y la tercera permite asignar el tamaño del atributo. Al ingresar el tamaño a cada atributo y hacer clic en el botón “Continue settings, continuar configuración” si los tamaños son correctos, se visualiza el mensaje “Tamaños agregados correctamente”, tal como se observa en la Figura 4.18.



Figura 4.18 Modal para la asignación de tamaño de atributos

El siguiente apartado de la Figura 4.17 muestra que la opción de CBIR por defecto aparece en “No”; esta alternativa solo se considera cuando la tabla o colección recibe consultas basadas en contenido, por medio de descriptores o datos multimedia, este trabajo no considera CBIR, por lo cual, se deja la opción en “No”.

Se sube el archivo de operaciones (*logs*), haciendo clic en la opción “*Choose, Escoger*”, se abrirá la ventana del gestor de archivos del sistema operativo, se busca el archivo y se selecciona para cargarlo en XAMANA, como se presenta en la Figura 4.19.

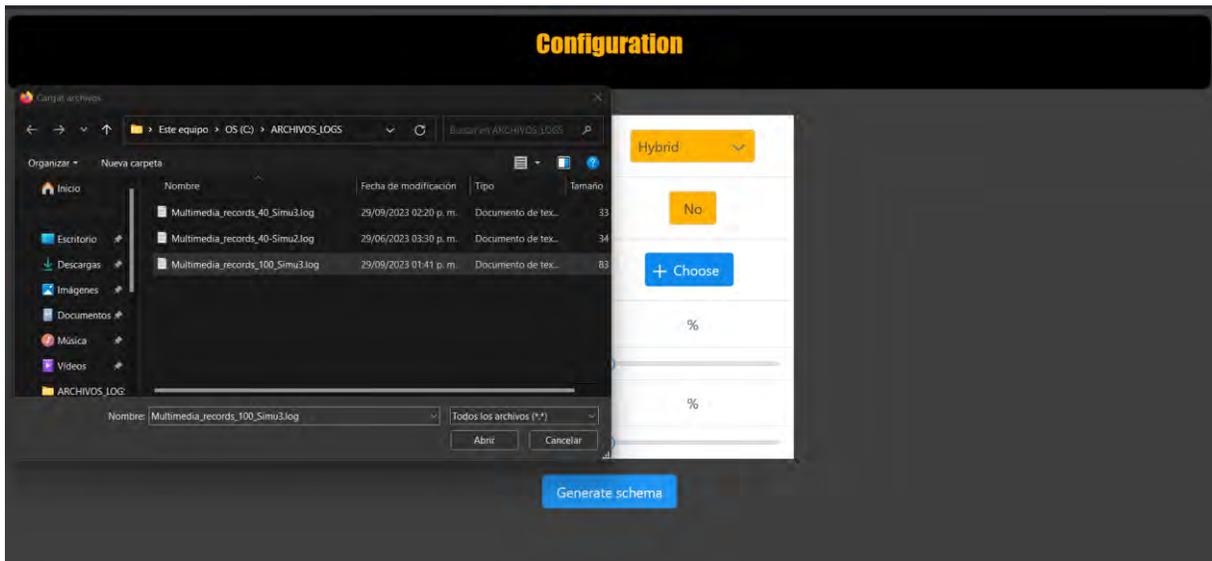


Figura 4.19 Escoger el archivo de operaciones (*logs*)

La carga de trabajo para la fragmentación híbrida estática se conforma de 100 operaciones, en la Tabla 4.1 se muestra la cantidad por tipo y en la Figura 4.20 se muestra un fragmento del *log*.

Tabla 4.1 Operaciones de la carga de trabajo para la fragmentación híbrida estática

Nombre del archivo	<i>Insert</i>		<i>Find</i>		<i>Update</i>		<i>Delete</i>	
multimedia_records_100_Simu3.log	23	Sitio 1:	25	Sitio 1:	30	Sitio 1:	22	Sitio 1:
		12		17		13		8
		Sitio 2:		Sitio 2:		Sitio 2:		Sitio 2:
		11		8		17		14

```

Multimedia_records_100_Simu3.log X
C:\Users\cresc> OneDrive > Escritorio > Multimedia_records_100_Simu3.log
1  :{"update":"multimedia_records","updates":[{"q":{"validado":{"$eq:null"},"u":{"$set":{"validado":null}},"multi":tr
2  "":{"find":"multimedia_records","filter":{"tipo":{"$eq:"personal"}}, "projection":{"validado":1},"lsid":{"id":{"$u
3  :{"delete":"multimedia_records","deletes":[{"q":{"descripcion":{"$eq:"."}}, "limit":0}], "ordered":true, "lsid":{"id
4  "":{"find":"multimedia_records","filter":{"validado":{"$ne:null}}, "projection":{"validado":1,"tipo":1},"lsid":{"i
5  :{"insert":"multimedia_records","documents":[{"_id":"LabIng06"},"ordered":true, "lsid":{"id":{"$uuid":"9e8b0ba
6  :{"insert":"multimedia_records","documents":[{"_id":"IMG1696"},"ordered":true, "lsid":{"id":{"$uuid":"9e8b0ba3-b5
7  :{"update":"multimedia_records","updates":[{"q":{"nombre":{"$eq:"ALICIA2"}}, "u":{"$set":{"nombre":"VELPERLE"}}, "mu
8  :{"update":"multimedia_records","updates":[{"q":{"descripcion":{"$eq:"."}}, "u":{"$set":{"descripcion":"..."}}, "mul
9  :{"delete":"multimedia_records","deletes":[{"q":{"validado":{"$eq:null}}, "limit":0}, "ordered":true, "lsid":{"id":{"
10 :{"insert":"multimedia_records","documents":[{"_id":"IMG385"},"ordered":true, "lsid":{"id":{"$uuid":"9e8b0ba3-b57
11 "":{"find":"multimedia_records","filter":{"nombre":{"$ne:"13MAY095"}}, "projection":{"descripcion":1},"lsid":{"id"
12 "":{"find":"multimedia_records","filter":{"descripcion":{"$eq:"."}}, "projection":{"tipo":1},"lsid":{"id":{"$uuid"
13 "":{"find":"multimedia_records","filter":{"descripcion":{"$eq:"Caricatura a mano"}}, "projection":{"nombre":1,"val
14 :{"update":"multimedia_records","updates":[{"q":{"_id":{"$eq:"IMG1842"}}, "u":{"$set":{"_id":"IMG402"}}, "multi":tru
15 :{"update":"multimedia_records","updates":[{"q":{"_id":{"$ne:"IMG1792"}}, "u":{"$set":{"_id":"IMG385"}}, "multi":tru
16 "":{"find":"multimedia_records","filter":{"validado":{"$ne:null}}, "projection":{"tipo":1,"nombre":1},"lsid":{"id"
17 "":{"find":"multimedia_records","filter":{"tipo":{"$eq:"personal"}}, "projection":{"_id":1,"tipo":1,"validado":1,"
18 :{"update":"multimedia_records","updates":[{"q":{"descripcion":{"$eq":"..."}}, "u":{"$set":{"descripcion":"..."}}, "mul
19 "":{"find":"multimedia_records","filter":{"validado":{"$ne:null}}, "projection":{"tipo":1,"_id":1,"descripcion":1,
20 "":{"find":"multimedia_records","filter":{"_id":{"$eq":{"$oid":"615e96ddf0f0e30bc46130cf"}}, "projection":{"_id":1
21 "":{"find":"multimedia_records","filter":{"nombre":{"$eq:"CREAS-3"}}, "projection":{"tipo":1,"descripcion":1,"vali
22 :{"insert":"multimedia_records","documents":[{"nombre":"Desfile1"},"ordered":true, "lsid":{"id":{"$uuid":"9e8b0ba
    
```

Figura 4.20 Fragmento del archivo log para realizar la fragmentación híbrida estática

El resultado obtenido en la aplicación Web XAMANA es una fragmentación estática, es decir, no es capaz de adaptarse a nuevas cargas de trabajo, por lo que, no es requerido asignar “*Operation threshold*, umbral de operaciones” y “*Performance threshold*, umbral de desempeño”, sin embargo, al trabajar con un esquema dinámico, es indispensable asignar estos dos valores (que van de 0 a 100), entonces, se asignan los umbrales moviendo de posición los *sliders* y se observa en la Figura 4.21; mientras más pequeño sea este valor es más sensible y son más frecuentes las refragmentaciones. El umbral de operaciones indica el número de operaciones que se requiere para resolver las operaciones al momento de fragmentar y el de desempeño si alcanzó el costo.

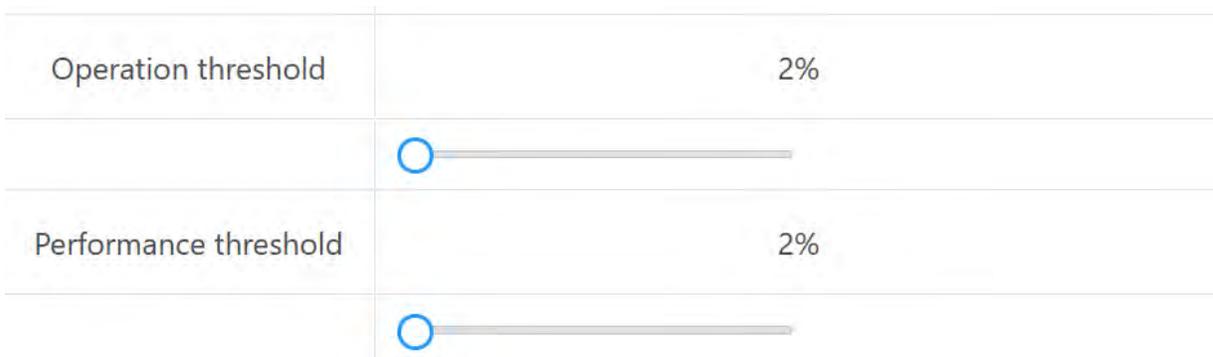


Figura 4.21 Asignación de umbral de operaciones y desempeño

En la Figura 4.22 se observa la configuración que el administrador de la base de datos ingresó para la fragmentación híbrida estática, entonces, el administrador hace clic en el botón “*Generate schema*, Generar esquema” y la aplicación muestra un modal con animación de carga, para indicar que se está generando el análisis y después mostrar el resultado de la fragmentación híbrida estática.

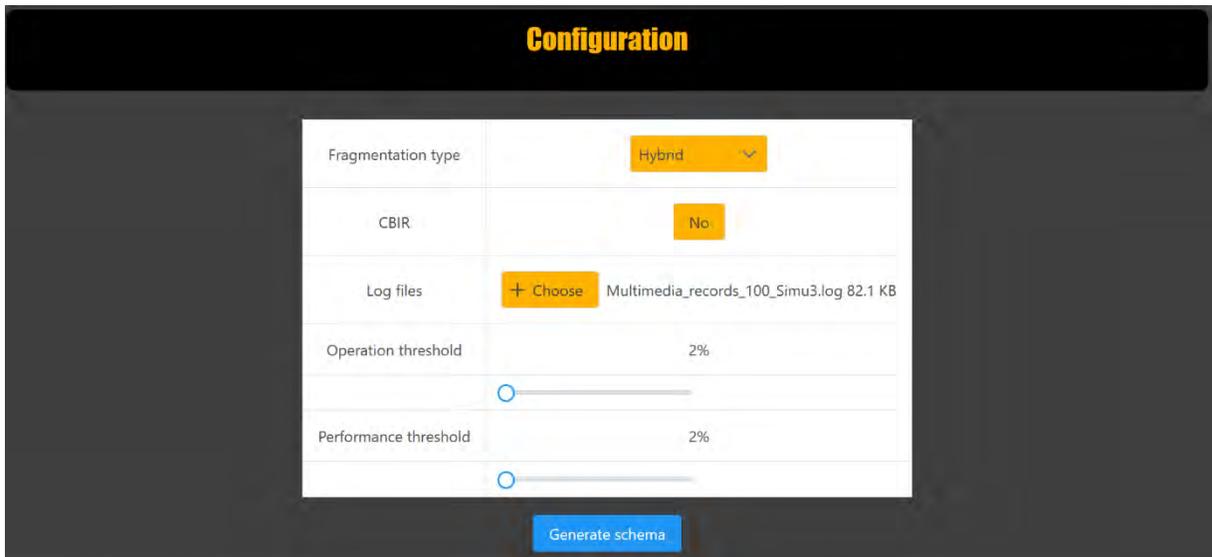


Figura 4.22 Configuración asignada por el DBA para fragmentación híbrida estática

Antes de mostrar el resultado final, se presenta el mensaje del esquema menos costoso como se visualiza en la Figura 4.23.

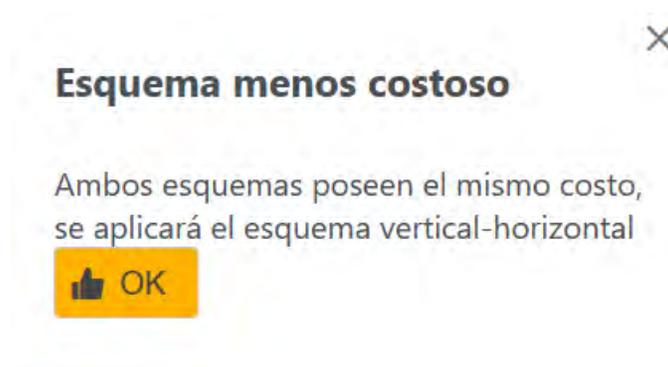


Figura 4.23 Mensaje con el esquema menos costoso

Finalmente, se obtiene el resultado (Figura 4.24) donde se presenta: el nombre, predicado con traslape, atributos, sitio horizontal y vertical, costo vertical y horizontal, por cada fragmento resultante.

Scheme						
Nombre	Predicado con traslape	Atributos	Sitio horizontal	Sitio vertical	Costo vertical	Costo horizontal
multimedia_records_1	{ "\$and": [{"validado": {"\$eq": null}}] }	descripcion,tipo,validado_id,	192.168.1.75	192.168.1.75	204610.8099999999	204610.8099999999
multimedia_records_2	{ "\$and": [{"validado": {"\$eq": null}}] }	imagen,descriptor_id	192.168.1.75	192.168.1.75	107764.22	107764.22
multimedia_records_3	{ "\$and": [{"or": [{"validado": {"\$not": {"\$eq": null}}}], [{"validado": {"\$ne": null}}] }	descripcion,tipo,validado_id,	192.168.1.75	192.168.1.75	142649.47999999995	142649.47999999995
multimedia_records_4	{ "\$and": [{"or": [{"validado": {"\$not": {"\$eq": null}}}], [{"validado": {"\$ne": null}}] }	imagen,descriptor_id	192.168.1.75	192.168.1.75	91449.98000000003	91449.98000000003

Costo HV: 205341.25999999983  
 Costo VH: 205341.25999999983  
 0%  
 Let's do it

Figura 4.24 Resultado final de la fragmentación híbrida estática

Para obtener el esquema de fragmentación híbrida estática se analizan las operaciones de la carga de trabajo, se crea el esquema vertical y horizontal, se mezclan y se obtiene el híbrido, a continuación, la Tabla 4.2 presenta el análisis de una operación seleccionada de las 100 operaciones presentadas en la Tabla 4.1.

Tabla 4.2 Ejemplo del análisis de costo de una operación para la fragmentación híbrida estática

Critero	Resultado
<b>Nro. de operación:</b> 3	db.multimedia_records.deleteMany({"descripcion": {"\$eq": "."}})
<b>Tipo y valor de la operación:</b>	Eliminación (delete) = 2
<b>Predicado de la operación:</b>	{"descripcion": {"\$eq": "."}}
<b>Número de tuplas por predicado:</b>	En 2 fragmentos se encontraron 1,296 tuplas que contienen el predicado = 2,592 tuplas.

Criterio	Resultado
	En 2 fragmentos se encontraron 9 tuplas que contienen el predicado = 18 tuplas. <b>Total de tuplas: 2,592 + 18 = 2,610</b>
<b>Costo horizontal:</b>	$Hor = tMax * VAO * VR$ $Hor = 2,610 * 2 * 2 = 10,440$
<b>Costo Vertical:</b>	$Ver = VAO * CR * VR * \sum_{i=1}^N Ta_i$ $Ver = 2 * 4 * 2 * 0.01 = 0.16$
<b>Costo híbrido:</b>	$Hib = \frac{(Hor+Ver)}{2}$ $Hib = (10,440 + 0.16) / 2$ <b>Hib = 5,220.08</b>

En la parte inferior de la página de la Figura 4.24 tiene el botón “Let’s do it, Hagámoslo”; para aplicar el esquema, el DBA hace clic sobre dicho botón. Se inicia la creación de cada fragmento en el sitio donde se requiere, se observa una barra de progreso en la parte inferior de la pantalla para esperar a que llegue al 100% y se muestre el *token*, Figura 4.25.

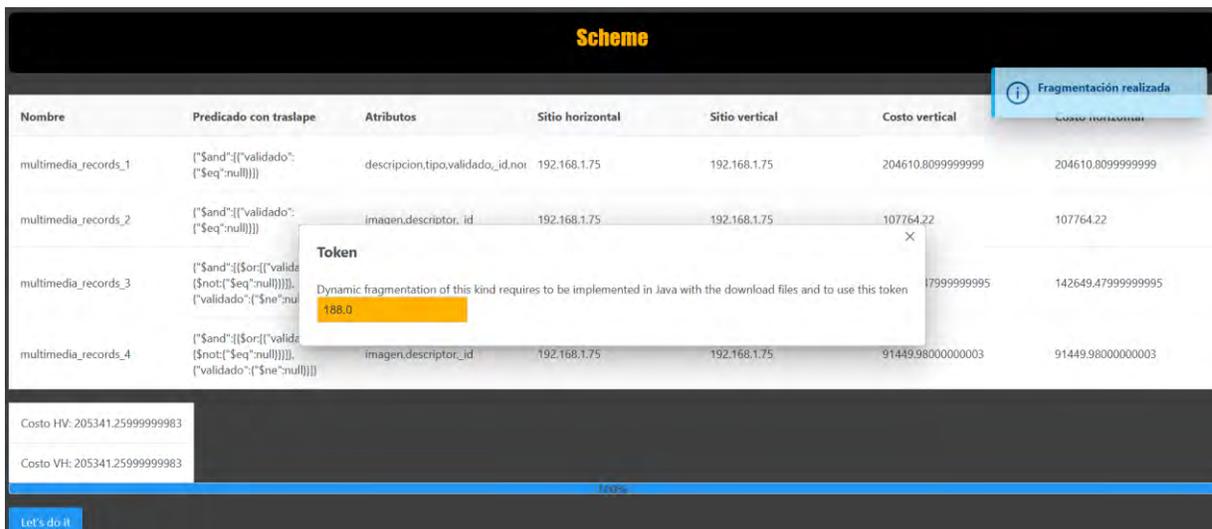


Figura 4.25 Aplicación de la fragmentación híbrida estática con el *token* correspondiente

El *token* es un valor entero, el cual, es el identificador de “tabla” de la base de datos XAMANA, que almacena la información de la partición realizada (nombre de la tabla fragmentada, umbral de operaciones y desempeño, tipo de fragmentación, identificador de la base de datos y el desempeño) que, a su vez, se relaciona con la tabla “fragmento” como se observa en la Figura 3.9.

Por lo tanto, al realizar la consulta: `SELECT * FROM TABLA JOIN FRAGMENTO ON FRAGMENTO.id_tabla=TABLA.id_tabla WHERE FRAGMENTO.id_tabla=188;` como se observa en la Figura 4.26, el resultado obtenido son los fragmentos híbridos estáticos, de tal modo que, el *token* es necesario para que el administrador de base de datos implemente la parte dinámica del método propuesto, en caso contrario, se mantendrá un esquema estático, como se mencionó anteriormente.

id_tabla	nombre	umbral_op	umbral_des	tipo_frag	id_BD	noperaciones
188	multimedia_records	2	2	c	188	100.000
188	multimedia_records	2	2	c	188	100.000
188	multimedia_records	2	2	c	188	100.000
188	multimedia_records	2	2	c	188	100.000

ndesempenio	id_fragmento	nombre	id_tabla	id_sitio	noperaciones	ndesempenio	isCBIR
205341.266	1054	multimedia_records_1	188	6	100.000	204610.812	0
205341.266	1055	multimedia_records_2	188	6	100.000	107764.219	0
205341.266	1056	multimedia_records_3	188	6	100.000	142649.484	0
205341.266	1057	multimedia_records_4	188	6	100.000	91449.977	0

Figura 4.26 Resultado de la consulta tabla *join* fragmento

A continuación, se presentan los resultados obtenidos al aplicar la fragmentación híbrida estática.

### 4.3.1 Resultado de la fragmentación híbrida estática

Los fragmentos resultantes de la partición híbrida, se muestran en la Figura 4.27, donde todos fueron asignados al sitio 2 (remoto), siendo un total de 4 particiones, mientras que, en el sitio 1 (local) desaparece la colección de *multimedia\_records* tal como se muestra en la Figura 4.28.

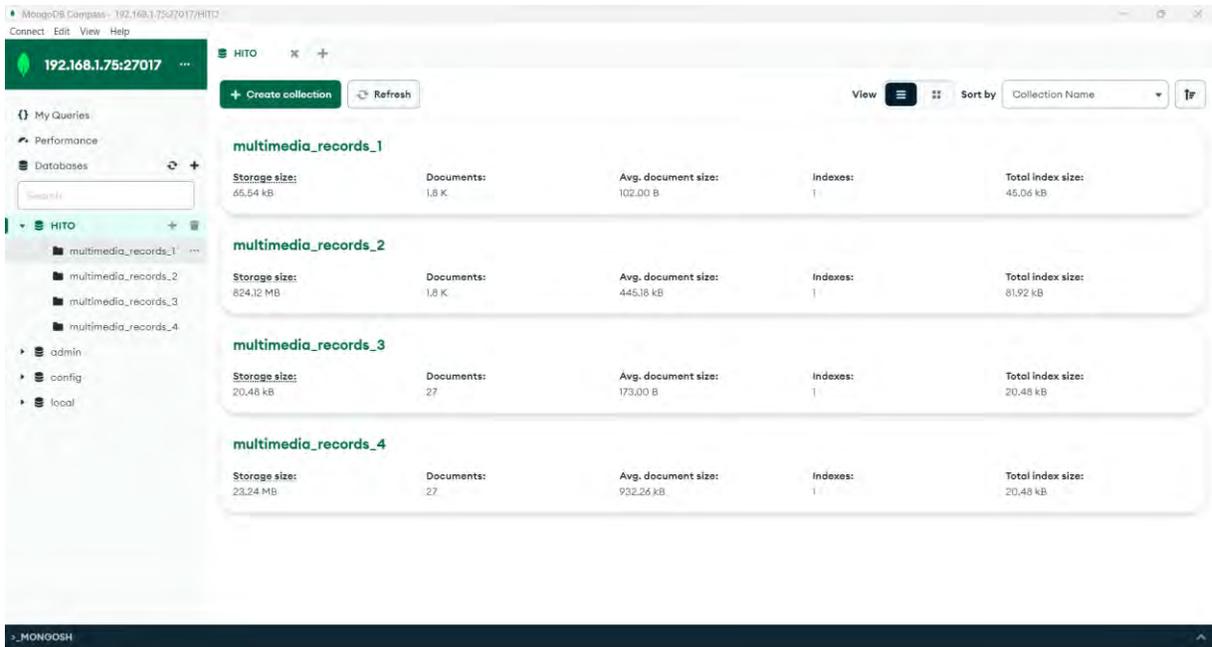


Figura 4.27 Fragmentos resultantes de la fragmentación híbrida estática en el sitio remoto

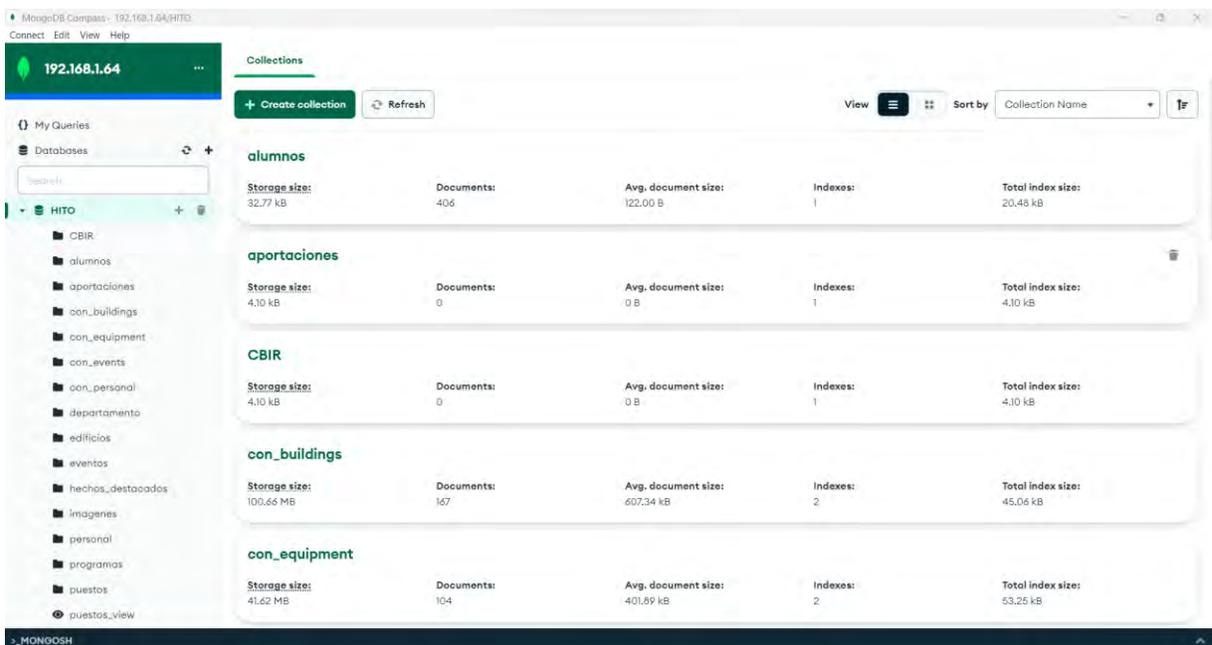


Figura 4.28 Colecciones de HITO en el sitio local ahora sin “*multimedia\_records*”

A continuación, en las Figuras 4.29 a 4.32 se presentan y describen cada uno de los fragmentos formados de manera más detalla, con el objetivo de tener más claro como se conforman los fragmentos antes de aplicar la fragmentación híbrida dinámica.

Como se observa en la Figura 4.29, el primer fragmento es *multimedia\_records\_1* y se conforma de: *\_id*, *nombre*, *descripción*, *validado* y *tipo* con un total de 1843 documentos, siendo: {"validado":{"\$eq":null}} el predicado que define este fragmento.

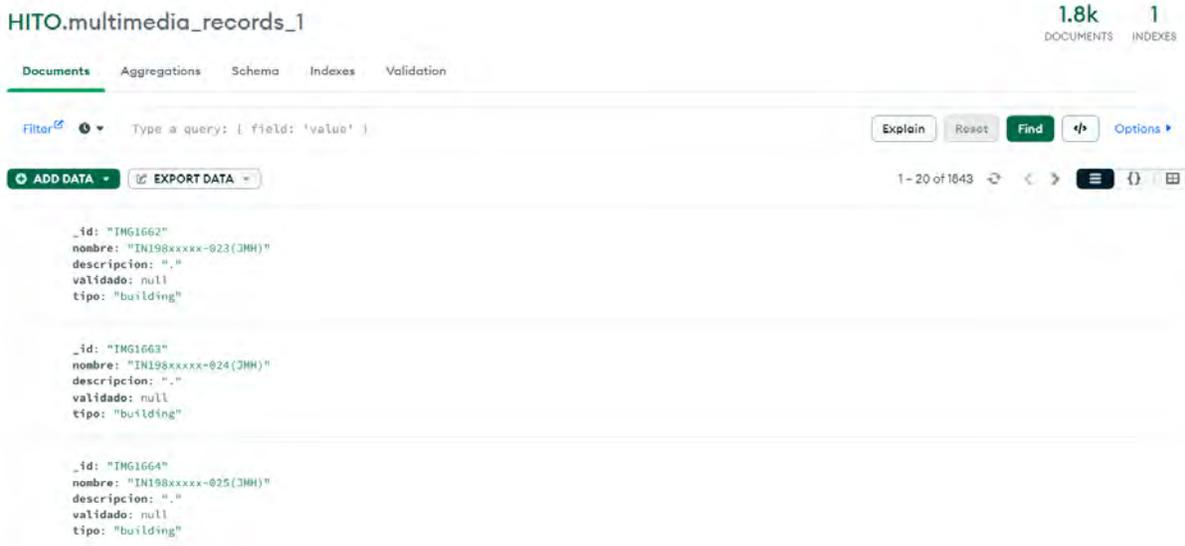


Figura 4.29 Fragmento “*multimedia\_records\_1*”

En *multimedia\_records\_2*, el predicado del fragmento es {"validado":{"\$eq":null}} y los atributos son: *\_id*, *imagen* y *descriptor* con 1843 documentos y se presenta en la Figura 4.30.

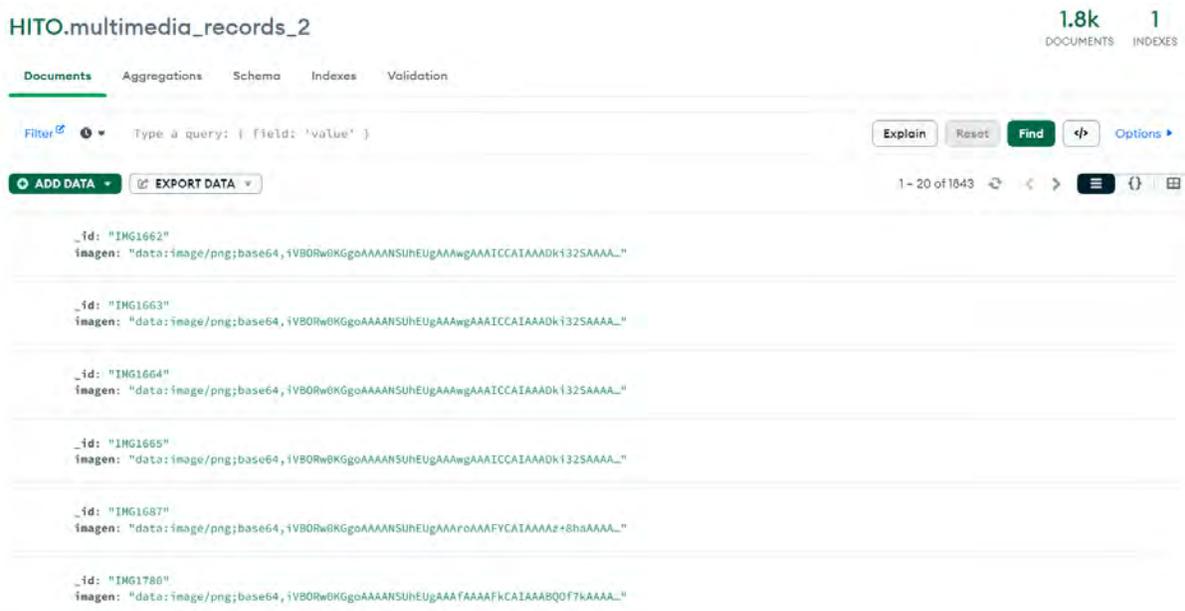


Figura 4.30 Fragmento “*multimedia\_records\_2*”

Mientras que, *multimedia\_records\_3* (Figura 4.31) y *multimedia\_records\_4* (Figura 4.32) constan de 27 documentos, para ambos el predicado que los define es {"validado":{"\$not":{"\$eq":null}}}, {"validado":{"\$ne":null}}, los atributos incluidos en *multimedia\_records\_3* son: *\_id*, *nombre*, *descripción*, *validado* y *tipo*; mientras que para *multimedia\_records\_4* los atributos que lo conforman son: *\_id*, *imagen* y *descriptor*.

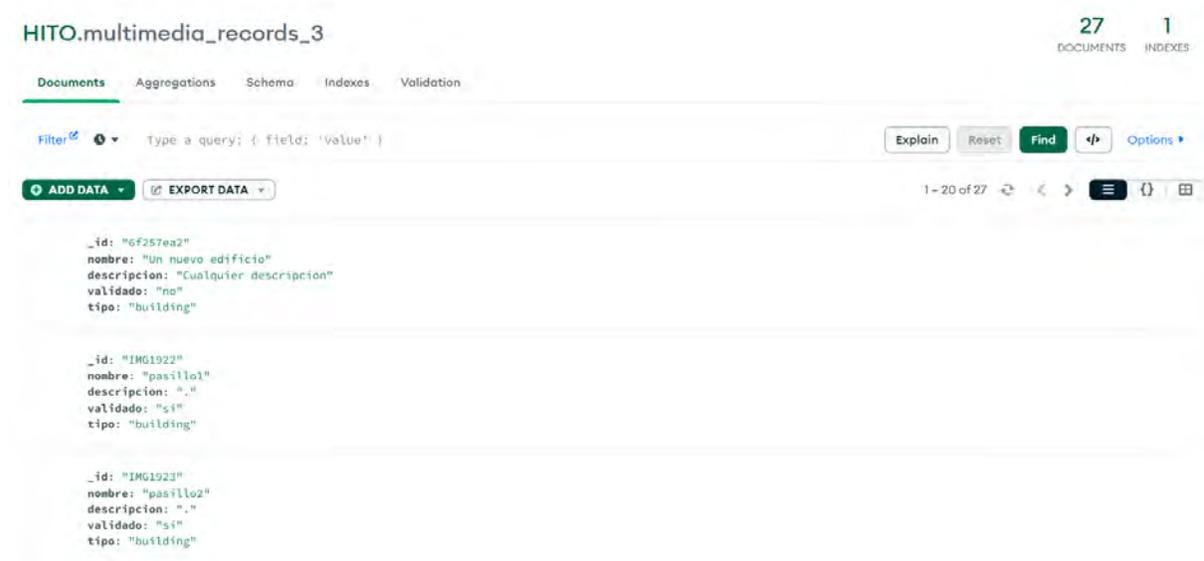


Figura 4.31 Fragmento “*multimedia\_records\_3*”

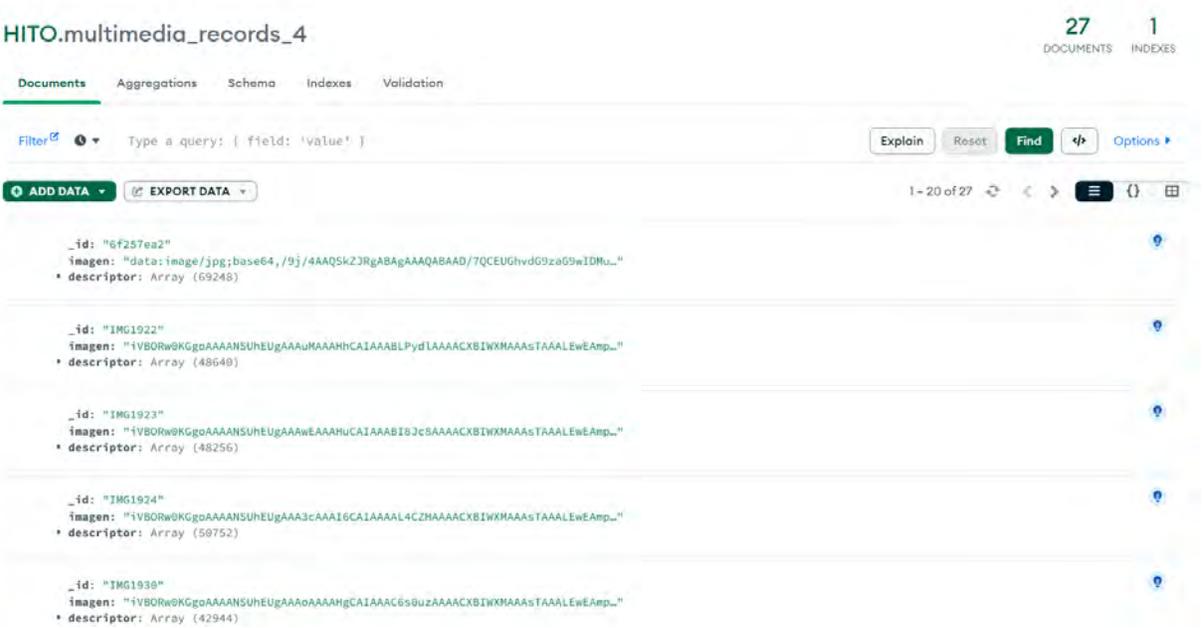


Figura 4.32 Fragmento “*multimedia\_records\_4*”

## 4.4 Vigilante-fragmentador

Para implementar el método híbrido dinámico propuesto en esta investigación, como ya se mencionó anteriormente, se consideraron dos sitios. Por consiguiente, en el primer sitio se utilizó el sistema operativo Windows 11 *Home Single Language* con un procesador AMD Ryzen 5 5600H con Radeon *Graphics* con una frecuencia de 3.30 GHz y 8 GB de RAM, mientras que, en el segundo sitio, Windows 10 Pro como sistema operativo, procesador AMD A9-9425 RADEON R5, 4 GB de RAM y con una frecuencia de 3.10 GHz. La base de datos centralizada (HITO) se encuentra en el sitio 1.

El vigilante-fragmentador debe estar en cada uno de los sitios que se contemplan junto con sus respectivas cargas de trabajo, de esta forma, solo re-fragmentará las particiones que se encuentren en su sitio (servidor) y cada vigilante-fragmentador evaluará si es necesario actualizar la información de los fragmentos que supervisa por cada sitio. En la Figura 4.33 se representa cómo funciona el vigilante-fragmentador, cada vigilante-fragmentador se comunica con una base de conocimientos y si es necesario, realizará el registro de un nuevo sitio o la actualización de fragmentos que se encuentren en cada sitio.

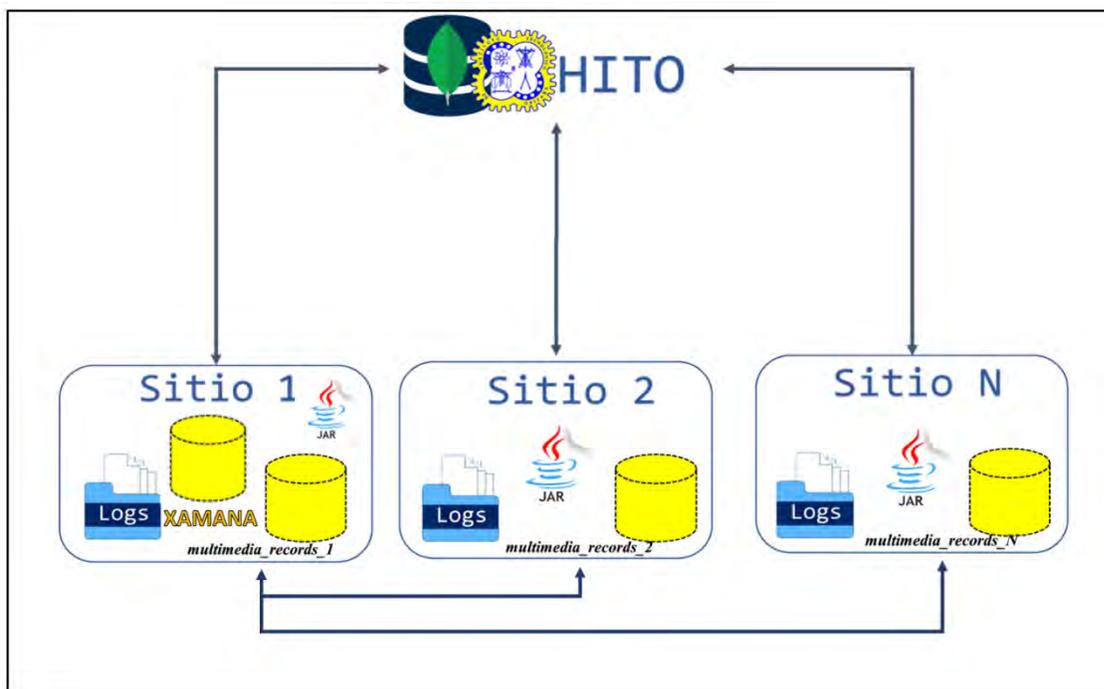


Figura 4.33 Vigilantes de fragmentación supervisando fragmentos por sitio

Como se mencionó en este capítulo, se tienen dos sitios, en el primer sitio que se considera como local (192.168.1.64) se encuentra la base de datos HITO con todas sus colecciones, pero sin contemplar a *multimedia\_records*, esta se eliminó al momento de aplicar la fragmentación híbrida estática porque fue la colección donde partió la creación de los fragmentos; en el otro sitio 192.168.1.75 (remoto) se formaron los nuevos fragmentos, por tal motivo, el vigilante-fragmentador se tiene que ejecutar en los dos sitios. Si existieran más sitios, es indispensable que el vigilante se ejecute en cada uno de ellos.

Al ejecutar el método, se mostrará la opción para ingresar el *token* (es un número entero que se obtiene al finalizar la fragmentación híbrida estática) y la ruta donde se encuentra la carga de trabajo de cada una de las operaciones de los fragmentos resultantes (Figura 4.34). Si el administrador de la base de datos ingresa el *token* erróneo, el vigilante indicará que es inválido, por otro lado, si la ruta que se ingresó no tiene la carga de trabajo mostrará que la carpeta no contiene archivos *logs*.

```
run:
Enter token obtained from the Web application to initiate the method (The token is only an integer)

Enter database log location: C:\LOGS
```

Figura 4.34 Inicio del vigilante-fragmentador

El umbral de operaciones es el número de operaciones que se requieren para llevar a cabo la re-fragmentación y para que se cumpla, el número de operaciones de cada fragmento tiene que ser mayor al valor del umbral de operaciones que el DBA ingresó. Con base en lo anterior, se presentan dos casos de estudio para la fragmentación híbrida dinámica. En el primero, todos los fragmentos (4) alcanzan el umbral de operaciones, esto es porque cada uno tiene 100 operaciones y se cumple por lo siguiente:  $100 \text{ operaciones} * 2 \text{ valor del umbral} * .01 = 2$ . Mientras que, en el segundo caso, solo tres fragmentos lo alcanzaron, esto es porque el último fragmento solo tiene una operación.

### 4.4.1 Fragmentación híbrida dinámica, primer caso de estudio

Para llevar a cabo la partición híbrida dinámica, se toman en cuenta los 4 fragmentos estáticos y por cada fragmento se realizan los 4 tipos de operaciones.

En la Figura 4.35 se presenta la carga de trabajo para realizar la fragmentación híbrida dinámica, que se conforma de 4 archivos *logs* y cada uno contiene 100 operaciones diferentes con los 4 tipos de operaciones (*insert*, *find*, *update*, *delete*) que corresponde a cada fragmento de la partición híbrida estática, siendo un total de 400 operaciones y se describen en la Tabla 4.3.

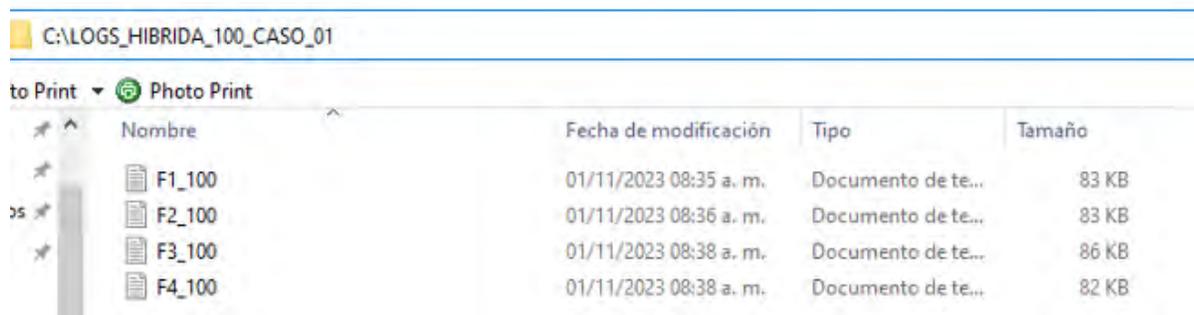


Figura 4.35 Carga de trabajo para la fragmentación híbrida dinámica, sitio remoto

Tabla 4.3 Operaciones de la carga de trabajo para la fragmentación híbrida dinámica, primer caso

Nombre del archivo	<i>Insert</i>		<i>Find</i>		<i>Update</i>		<i>Delete</i>	
	Sitio 1:	Sitio 2:	Sitio 1:	Sitio 2:	Sitio 1:	Sitio 2:	Sitio 1:	Sitio 2:
F1_100	11	19	18	6	12	14	9	11
	30	24	26	20				
F2_100	14	11	8	14	15	12	13	13
	25	22	27	26				
F3_100	3		15		21		11	

Nombre del archivo	<i>Insert</i>		<i>Find</i>		<i>Update</i>		<i>Delete</i>	
	<b>14</b>	Sitio 2: <b>11</b>	<b>27</b>	Sitio 2: <b>12</b>	<b>37</b>	Sitio 2: <b>16</b>	<b>22</b>	Sitio 2: <b>11</b>
F4_100	<b>29</b>	Sitio 1: <b>15</b>	<b>19</b>	Sitio 1: <b>11</b>	<b>24</b>	Sitio 1: <b>9</b>	<b>28</b>	Sitio 1: <b>15</b>
		Sitio 2: <b>14</b>		Sitio 2: <b>8</b>		Sitio 2: <b>15</b>		Sitio 2: <b>13</b>

La Figura 4.36 muestra un fragmento del primer archivo *log* que corresponde a las operaciones de *multimedia\_records\_fl*; como ya se presentó en la Tabla 4.3, este contiene 30 inserciones, 20 eliminaciones, 26 actualizaciones y 24 consultas, siendo un total de 100 operaciones.

```

F1_100.log X
C: > LOGS_HIBRIDA_100_CASO_01 > F1_100.log
1  {"insert":"multimedia_records_1","documents":[{"descripcion":"."},"ordered":true,"lsid":{"id":{"$uuid":"9e8b0b
2  {"delete":"multimedia_records_1","deletes":[{"q":{"descripcion":{"$eq":"."},"limit":0},"ordered":true,"lsid":{"
3  {"update":"multimedia_records_1","updates":[{"q":{"validado":{"$eq":null},"u":{"$set":{"validado":null}},"multi
4  :{"find":"multimedia_records_1","filter":{"tipo":{"$eq:"personal"}},"projection":{"_id":1},"lsid":{"id":{"$uui
5  :{"find":"multimedia_records_1","filter":{"_id":{"$ne":{"$oid":"615e99aef0f0e30bc4613360"}}},"projection":{"nom
6  {"insert":"multimedia_records_1","documents":[{"validado":null},"ordered":true,"lsid":{"id":{"$uuid":"9e8b0ba3
7  :{"find":"multimedia_records_1","filter":{"descripcion":{"$eq:"Caricatura a mano"}},"projection":{"descripcion
8  {"delete":"multimedia_records_1","deletes":[{"q":{"validado":{"$eq":null},"limit":0},"ordered":true,"lsid":{"id
9  {"insert":"multimedia_records_1","documents":[{"_id":"IMG388"},"ordered":true,"lsid":{"id":{"$uuid":"9e8b0ba3-
10 :{"find":"multimedia_records_1","filter":{"nombre":{"$eq:"LEVYARRA"}},"projection":{"tipo":1,"descripcion":1},
11 :{"find":"multimedia_records_1","filter":{"tipo":{"$ne:"personal"}},"projection":{"_id":1},"lsid":{"id":{"$uui
12 {"insert":"multimedia_records_1","documents":[{"descripcion":"."},"ordered":true,"lsid":{"id":{"$uuid":"9e8b0b
13 {"update":"multimedia_records_1","updates":[{"q":{"nombre":{"$eq:"EN198xxxx-017(JMH)"},"u":{"$set":{"nombre":"
14 :{"find":"multimedia_records_1","filter":{"descripcion":{"$eq":"."},"projection":{"nombre":1,"_id":1},"lsid":{"
15 :{"find":"multimedia_records_1","filter":{"nombre":{"$eq:"EN198xxxx-014(JMH)"},"projection":{"_id":1,"descri
16 {"insert":"multimedia_records_1","documents":[{"validado":null},"ordered":true,"lsid":{"id":{"$uuid":"9e8b0ba3
17 :{"find":"multimedia_records_1","filter":{"validado":{"$ne:null}}},"projection":{"nombre":1,"descripcion":1},"l
18 :{"find":"multimedia_records_1","filter":{"validado":{"$ne:null}}},"projection":{"_id":1},"lsid":{"id":{"$uui
19 :{"find":"multimedia_records_1","filter":{"validado":{"$ne:null}}},"projection":{"tipo":1},"lsid":{"id":{"$uui
20 :{"find":"multimedia_records_1","filter":{"_id":{"$ne":{"$oid":"615e975ef0f0e30bc4613113"}}},"projection":{"_id
21 {"insert":"multimedia_records_1","documents":[{"tipo":"event"},"ordered":true,"lsid":{"id":{"$uuid":"9e8b0ba3-
22 {"delete":"multimedia_records_1","deletes":[{"q":{"tipo":{"$ne:"building"},"limit":0},"ordered":true,"lsid":{"
  
```

Figura 4.36 Fragmento del primer archivo *log* de “*multimedia\_records\_1*”

En el sitio 1 (local) donde aún no hay fragmentos, se tiene un archivo *log* en blanco (Figura 4.37), cuando existan fragmentos en el sitio local, se agregarán las operaciones en el archivo que se ingresó y el vigilante automáticamente las detectará y las considerará para su análisis.

El caso contrario sucede en el sitio remoto, donde se tiene una carpeta con cada una de las cargas de trabajo para cada fragmento, como ya se explicó anteriormente.

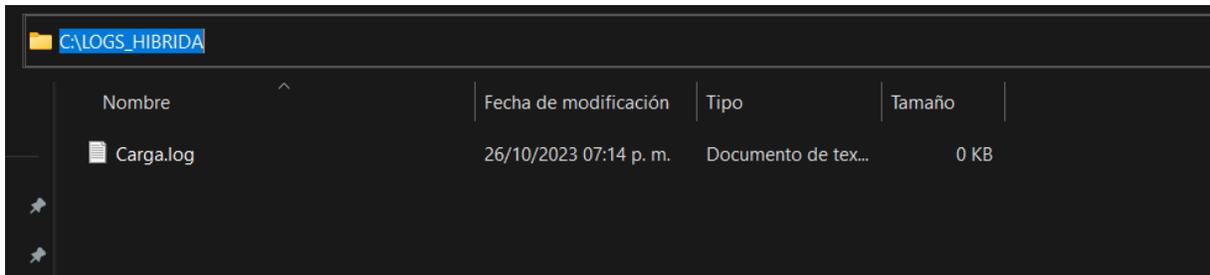


Figura 4.37 Carga de trabajo para la fragmentación híbrida dinámica, sitio local

Las Figuras 4.38 y 4.39 muestran la ejecución del vigilante-fragmentador en diferentes sitios; en la primera, el vigilante se ejecuta en el sitio 2 donde se encuentran los fragmentos producto de la fragmentación híbrida estática, mientras que, en la segunda figura, el vigilante está en el sitio 1 donde no se tienen fragmentos de *multimedia\_records*.

```

Enter token obtained from the Web application to initiate the method (The token is only an integer)
188
Enter database log location: C:\LOGS
C:\LOGS_HIBRIDA_100_CASO_01

-----
Data found and information obtained
-----
Database name           HITO
Database IP             192.168.1.64
Fragmented table        multimedia_records
Operation threshold     2.0
Performance threshold  2.0
Total operation log     100.0
-----

Fragments sites finds
Fragment name: multimedia_records_1
Fragment name: multimedia_records_2
Fragment name: multimedia_records_3
Fragment name: multimedia_records_4
Fragments scheme found
Accepting connections over socket
    
```

Figura 4.38 Vigilante-fragmentador, sitio remoto

```

run:
Enter token obtained from the Web application to initiate the method (The token is only an integer)
188
Enter database log location: C:\LOGS
C:\\LOGS_HIBRIDA

-----
Data found and information obtained
-----
Database name           HITO
Database IP             192.168.1.64
Fragmented table       multimedia_records
Operation threshold     2.0
Performance threshold  2.0
Total operation log     100.0
-----

Fragments scheme found
Fragments remote sites finds
Fragment name: multimedia_records_1
Fragment name: multimedia_records_2
Fragment name: multimedia_records_3
Fragment name: multimedia_records_4
Fragments scheme found
Accepting connections over socket

```

Figura 4.39 Vigilante-fragmentador, sitio local

Al ingresar el *token* (188) se muestra la información del nombre de la base de datos, la dirección IP, tabla y valores de los umbrales (operación y desempeño) que se encuentran almacenados en la base de datos XAMANA y que corresponden a la fragmentación híbrida estática. Es necesario que el administrador de la base de datos verifique el *token* y la ruta de la carga de trabajo, esto es porque existen múltiples esquemas de fragmentación de otros usuarios en la base de datos y es necesario asegurarse que el vigilante-fragmentador esté supervisando el esquema deseado.

Con base en el Capítulo 3, sección 3.3.2.4 modelo de proceso y lo mostrado en la Figura 3.15 (Diagrama de proceso de la fragmentación híbrida), se inicia supervisando qué fragmentos alcanzaron el umbral de operaciones, si esto se cumple, se realiza el segundo paso, el cual es, analizar el costo del esquema actual. En la Figura 4.40, se observa que los cuatro fragmentos alcanzaron el umbral de operaciones, por lo tanto, el vigilante-fragmentador lee las operaciones realizadas a cada uno de los fragmentos y lleva a cabo el análisis del costo del esquema. Al no poseer fragmentos y operaciones, la consola del sitio 1 no presenta ningún análisis, como se observa en la Figura 4.41. La Figura 4.42 describe el análisis de dos operaciones seleccionadas del fragmento *multimedia\_records\_1*, como ya se mencionó se realizará con los fragmentos restantes.

```

-----
Current operation information for hybrid fragmentation (first analysis):
-----
Fragment name:          multimedia_records_1
Achieve Operation:      2.0
Number of Operations:   100
-----

Current operation information for hybrid fragmentation (first analysis):
-----
Fragment name:          multimedia_records_2
Achieve Operation:      2.0
Number of Operations:   100
-----

Current operation information for hybrid fragmentation (first analysis):
-----
Fragment name:          multimedia_records_3
Achieve Operation:      2.0
Number of Operations:   100
-----

Current operation information for hybrid fragmentation (first analysis):
-----
Fragment name:          multimedia_records_4
Achieve Operation:      2.0
Number of Operations:   100
-----

```

Figura 4.40 Fragmentos que alcanzaron el umbral de operaciones, primer caso

```

-----The file has finished analyze -----
Accepting connections over socket

-----HYBRID COST-----
-----The file has finished analyze -----

-----HYBRID COST-----
-----The file has finished analyze -----

-----HYBRID COST-----
-----The file has finished analyze -----

```

Figura 4.41 Análisis de operaciones del vigilante-fragmentador, sitio local

```

Number of operation: 11
{"t":{"$date":"2023-09-27T09:11:52.224000Z"},"s":"I", "c":"COMMAND", "id":1262, "ctx":"conn1","msg":"Slow query","attr":{"type":"command"},
Count of tuples: {"tipo":{"$ne:"personal"}}
Number of tuples: 972.0

Count of tuples: {"tipo":{"$ne:"personal"}}
Number of tuples: 1843.0

Count of tuples: {"tipo":{"$ne:"personal"}}
Number of tuples: 24.0

Count of tuples: {"tipo":{"$ne:"personal"}}
Number of tuples: 27.0

VAO: 2
CR: 4

Cost of vertical operation: 0.16
Cost of horizontal operation: 5732.0
Cost of hybrid operation: 2866.08
The cost of operation is allocated to the fragment: multimedia_records_1
The cost of operation is allocated to the fragment: multimedia_records_2
The cost of operation is allocated to the fragment: multimedia_records_3
The cost of operation is allocated to the fragment: multimedia_records_4

Number of operation: 50
{"t":{"$date":"2023-09-27T09:11:53.102000Z"},"s":"I", "c":"COMMAND", "id":1262,"ctx":"conn1","msg":"Slow query","attr":{"type":"command"},
Count of tuples: {"descripcion":{"$eq":"."}}
Number of tuples: 1296.0

Count of tuples: {"descripcion":{"$eq":"."}}
Number of tuples: 9.0

VAO: 6
CR: 2

Cost of vertical operation: 0.12
Cost of horizontal operation: 7830.0
Cost of hybrid operation: 3915.06
The cost of operation is allocated to the fragment: multimedia_records_1
The cost of operation is allocated to the fragment: multimedia_records_3

```

Figura 4.42 Análisis de operaciones del vigilante-fragmentador, sitio remoto

A continuación, en la Tabla 4.4 se explica con más detalle el cálculo de las operaciones 11 y 50 seleccionadas del archivo *log* del primer fragmento (*multimedia\_records\_1*) y que corresponden a las mostradas en la Figura 4.42; se analizan más operaciones (400), pero solo se presentan dos como ejemplo.

Tabla 4.4 Análisis de costo de operaciones en *multimedia\_records\_1*, fragmentación híbrida dinámica, primer caso

Criterio	Resultado
<b>Nro. de operación:</b>  <b>11</b>	db.multimedia_records.find({"tipo": {"\$ne:"personal"} }, {"_id":1})

Criterio	Resultado
Tipo y valor de la operación:	Lectura ( <i>find</i> ) = <b>1</b>
Predicado de la operación:	{"tipo": {\$ne:"personal"}}
VAO:	2 (valor remoto) * 1 del valor de operación = <b>2</b>
CR:	<b>4</b> fragmentos que se requieren para la fragmentación
Número de tuplas por predicado:	En 1 fragmento se encontraron 972 tuplas que contienen el predicado En 1 fragmento se encontraron 1843 tuplas que contienen el predicado En 1 fragmento se encontraron 24 tuplas que contienen el predicado En 1 fragmento se encontraron 27 tuplas que contienen el predicado <b>Total de tuplas: 972+1843+24+27 = 2,866</b>
Costo horizontal:	$Hor = tMax * VAO * VR$ $Hor = 2,866 * 2 * 1 = \mathbf{5,732}$
Costo Vertical:	$Ver = VAO * CR * VR * \sum_{i=1}^N Ta_i$ $Ver = 1 * 4 * 2 * 0.02 = \mathbf{0.16}$
Costo híbrido:	$Hib = \frac{(Hor+Ver)}{2}$ $Hib = (5,732 + 0.16) / 2$ $Hib = \mathbf{2866.08}$
Nro. de operación: 50	db.multimedia_records.updateMany({"descripcion":{"\$eq:"."}}, {"\$set":{"descripcion":"Caricatura a mano"}})
Tipo y valor de la operación:	Actualización ( <i>update</i> ) = <b>3</b>

<b>Criterio</b>	<b>Resultado</b>
<b>Predicado de la operación:</b>	$\{"descripcion":\{\$eq:". "\}\}$
<b>VAO:</b>	2 (valor remoto) * 3 del valor de operación = <b>6</b>
<b>CR:</b>	2 fragmentos que se requieren para la fragmentación
<b>Número de tuplas por predicado:</b>	En 1 fragmento se encontraron 1,296 tuplas que contienen el predicado En 1 fragmento se encontraron 9 tuplas que contienen el predicado <b>Total de tuplas: 1,296 + 9 = 1,305</b>
<b>Costo horizontal:</b>	$Hor = tMax * VAO * VR$ $Hor = 1,305 * 3 * 2 = \mathbf{7,830}$ tuplas
<b>Costo Vertical:</b>	$Ver = VAO * CR * VR * \sum_{i=1}^N Ta_i$ $Ver = 3 * 2 * 1 * 0.02 = \mathbf{0.12}$
<b>Costo híbrido:</b>	$Hib = \frac{(Hor+Ver)}{2}$ $Hib = (7,830 + 0.12) / 2$ $Hib = \mathbf{3,915.06}$

Tal como se indica en la Figura 3.15, cuando el vigilante-fragmentador termina de analizar el costo de cada fragmento, detectará qué fragmentos alcanzaron el umbral de desempeño. Si alcanzaron el umbral, se leen nuevamente las operaciones, pero ahora para crear el esquema de partición híbrida, iniciando con el vertical y después el horizontal.

En la Figura 4.43 se presenta la información de costos del primer fragmento analizado “*multimedia\_records\_1*”, este proceso se repite con el resto de los fragmentos (*multimedia\_records\_2*, *multimedia\_records\_3* y *multimedia\_records\_4*).

```
-----
Current cost information for hybrid fragmentation (second analysis):
-----
Cost in multimedia_records_1: 754806.7300000011
Performance threshold: 2.0
Initial performance: 204610.812
Achieve performance: 4092.21624
-----

This fragment is not CBIR

-----CREATE HYBRID SCHEME-----
```

Figura 4.43 Información del costo del fragmento “*multimedia\_records\_1*”, primer caso

Como se observa en la Tabla 4.5, el resto de los fragmentos (incluyendo *multimedia\_records\_1*) tienen diferentes costos y todos alcanzaron el umbral de desempeño. Por cada fragmento, se crea el esquema de fragmentación vertical y horizontal, posteriormente, se crea el nuevo esquema de fragmentación híbrida.

Tabla 4.5 Comparación de los costos de los fragmentos híbridos, primer caso

Nombre del fragmento	Costo	Umbral de desempeño	Rendimiento inicial	Rendimiento a alcanzar
<i>multimedia_records_2</i>	577,203.730	2	107,764.219	2,155.284
<i>multimedia_records_3</i>	700,075.130		142,649.484	2,852.989
<i>multimedia_records_4</i>	577,123.960		91,449.977	1,828.999

En la Figura 4.44 se muestra el resultado final del esquema de partición vertical y horizontal contemplando a *multimedia\_records\_1* posteriormente, en la Tabla 4.6 se compara el resto de los fragmentos verticales y horizontales formados en la partición dinámica, antes de crear el esquema híbrido. La primera columna de la Tabla 4.6 indica el número de fragmento del cual se derivó, es decir, el 2 indica que el esquema partió de *multimedia\_records\_2* y así sucesivamente, la segunda indica el nombre que se le asignó al fragmento, la tercera muestra los atributos de los fragmentos, las filas que contienen una línea punteada es porque no se formó el fragmento vertical, y la última, los predicados de la fragmentación horizontal.

```

Vertical scheme
Fragment name: multimedia records 1
Attributes: descripcion
Attributes: tipo
Attributes: validado
Attributes: _id
Attributes: nombre

Horizontal scheme
Fragment name: multimedia_records_1
Predicate: {"validado":{"$eq":null}}
Overlap: {"$and":[{"validado":{"$eq":null}}]}

Horizontal scheme
Fragment name: multimedia_records_3
Predicate: {"validado":{"$ne":null}}
Overlap: {"$and":[{"$or":[{"validado":{"$not":{"$eq":null}}}],{"validado":{"$ne":null}}]}
    
```

Figura 4.44 Esquema vertical y horizontal dinámico considerando “*multimedia\_records\_1*”, primer caso

Tabla 4.6 Comparación de los fragmentos verticales y horizontales en la fragmentación dinámica, primer caso de estudio

Nro. de fragmento	Nombre del fragmento	Esquema Vertical	Esquema Horizontal
2	<i>multimedia_records_1</i>	<i>imagen descriptor _id</i>	<b>Predicado:</b> {"_id":{"\$ne":"IMG1631"}} <b>Traslape:</b> {"\$and":[{"_id":{"\$ne":"IMG1631"}}]}
	<i>multimedia_records_2</i>	-----	<b>Predicado:</b> {"_id":{"\$ne":{"\$oid":"615e9812f0f0e30bc46131b5"}}} <b>Traslape:</b> {"\$and":[{"\$or":[{"_id":{"\$not":{"\$ne":"IMG1631"}}}],{"_id":{"\$ne":{"\$oid":"615e9812f0f0e30bc46131b5"}}}]}}
	<i>multimedia_records_1</i>	<i>descripción tipo validado</i>	<b>Predicado:</b> {"tipo":{"\$ne":"building"}}}

Nro. de fragmento	Nombre del fragmento	Esquema Vertical	Esquema Horizontal
3		<i>_id</i> <i>nombre</i>	<b>Traslape:</b> { "\$and": [ { "tipo": { "\$ne": "building" } } ] }
	<i>multimedia_records_2</i>	-----	<b>Predicado:</b> { "tipo": { "\$ne": "event" } } <b>Traslape:</b> { "\$and": [ { \$or: [ { "tipo": { \$not: { "\$ne": "building" } } ] } ] }, { "tipo": { "\$ne": "event" } } ] }
4	<i>multimedia_records_1</i>	<i>imagen</i> <i>descriptor</i> <i>_id</i>	<b>Predicado:</b> { "_id": { "\$ne": "IMG841" } } <b>Traslape:</b> { "\$and": [ { "_id": { "\$ne": "IMG841" } } ] }
	<i>multimedia_records_2</i>	-----	<b>Predicado:</b> { "_id": { "\$ne": "IMG1924" } } <b>Traslape:</b> { "\$and": [ { \$or: [ { "_id": { \$not: { "\$ne": "IMG841" } } ] } ] }, { "_id": { "\$ne": "IMG1924" } } ] }

Se obtiene el nuevo esquema de partición híbrida dinámica, por cada fragmento de la fragmentación híbrida estática. Hay dos criterios que se consideran antes de refragmentar y asignar:

- 1) Cuando uno de los fragmentos híbridos tiene el mismo número de tuplas y atributos del fragmento original, no se realiza la partición nuevamente, pero si cambia el sitio donde debe asignarse, entonces este se mueve de su lugar de origen, es decir, se reasigna al sitio destino y se conserva como está formado, manteniendo su nombre.
- 2) Si el fragmento híbrido está vacío (no contiene tuplas, atributos), entonces no se forma y mucho menos se nombra.

Por cada esquema de fragmentación híbrida se presenta su información antes de aplicarse: número de tuplas del anterior y nuevo fragmento, nombre del fragmento, predicado (fragmentación horizontal), atributos (fragmentación vertical), costo horizontal-vertical y vertical-horizontal. En la Figura 4.45 se muestra lo descrito anteriormente, sin embargo, el nuevo esquema de fragmentación cumple con los dos criterios mencionados y no se vuelve a presentar la información del fragmento (nombre, predicado y atributos) porque es igual al original; solo se muestra cuando sucede lo contrario, es decir, los atributos y el número de tuplas son diferentes al fragmento original como ocurre en *multimedia\_records\_2\_1* y *multimedia\_records\_2\_2* (Figura 4.46).

```

-----
Hybrid Scheme
-----
Total tuples of the original fragment: 1843.0
Total tuples of the new fragment: 1843
Do not required fragmentation because have equals attributes and tuples
Total tuples of the original fragment: 1843.0
Total tuples of the new fragment: 0
The fragment is empty
Total tuples of the original fragment: 1843.0
Total tuples of the new fragment: 0
The fragment is empty
H-V cost: 264566.76
V-H cost: 264566.76
-----
Both schemes have equals cost but this will be applied vertical-horizontal scheme

```

Figura 4.45 Esquema de la fragmentación híbrida dinámica “*multimedia\_records\_1*”, primer caso

```

-----
Hybrid Scheme
-----
Total tuples of the original fragment: 1843.0
Total tuples of the new fragment: 1842
Fragment name: multimedia_records_2_1
Overlap: {"$and":[{"_id":{"$ne":"IMG1631"}}]}
Attribute: imagen
Attribute: descriptor
Attribute: _id

Total tuples of the original fragment: 1843.0
Total tuples of the new fragment: 1
Fragment name: multimedia_records_2_2
Overlap: {"$and":[{"$or":[{"_id":{"$not":{"$ne":"IMG1631"}}}],{"_id":{"$ne":{"$oid":"615e9812f0f0e30bc46131b5"}}]}]}
Attribute: imagen
Attribute: descriptor
Attribute: _id
H-V cost: 157170.17000000004
V-H cost: 157170.17000000004
-----
Both schemes have equals cost but this will be applied vertical-horizontal scheme

```

Figura 4.46 Esquema de la fragmentación híbrida dinámica “*multimedia\_records\_2\_1*” y “*multimedia\_records\_2\_2*”, primer caso

Finalmente, el nuevo esquema de fragmentación híbrida se asigna y aplica al sitio correspondiente (sitio 1), se actualiza la información en la base de datos XAMANA (Figura 4.47 y 4.48) y el vigilante del sitio 1 (local) detecta que se ha realizado una refragmentación y este actualiza su información, como se observa en la Figura 4.49. Todos los vigilantes actualizan su información cuando detectan una nueva refragmentación sin importar donde se encuentren (sitio).

```
-----
Apply new Scheme Hybrid
-----
update Fragmento set id_sitio = '5.0' where id_fragmento = 1054.0
Deleting origin.....
-----
```

Figura 4.47 Asignar el nuevo fragmento híbrido dinámico “*multimedia\_records\_1*”, primer caso

```
-----
Apply new Scheme Hybrid
-----
Fragment name: multimedia_records_2_1
Destination: 192.168.1.64
insert into Fragmento(nombre, id_tabla, id_sitio, noperaciones, ndesempenio, isCBIR) value ('multimedia_records_2
Fragment name: multimedia_records_2_2
Destination: 192.168.1.64
insert into Fragmento(nombre, id_tabla, id_sitio, noperaciones, ndesempenio, isCBIR) value ('multimedia_records_2
Deleting origin.....
-----
```

Figura 4.48 Asignar el nuevo fragmento híbrido dinámico “*multimedia\_records\_2\_1*” y “*multimedia\_records\_2\_2*”, primer caso

```
Re-fragment flag: true
Accepting connections over socket

Fragments sites finds
Fragment name: multimedia_records_1
Fragments remote sites finds
Fragment name: multimedia_records_2
Fragment name: multimedia_records_3
Fragment name: multimedia_records_4

Re-fragment flag: true
Accepting connections over socket
Re-fragment flag: true
Accepting connections over socket

Fragments sites finds
Fragment name: multimedia_records_1
Fragment name: multimedia_records_2_1
Fragment name: multimedia_records_2_2
Fragments remote sites finds
Fragment name: multimedia_records_3
Fragment name: multimedia_records_4
```

Figura 4.49 Actualización de vigilante en sitios, primer caso

El resultado obtenido en la fragmentación híbrida dinámica son 7 fragmentos que se derivaron del esquema de partición estática como ya se ha mencionado; en la Tabla 4.7 se detallada cada fragmento formado.

Al terminar de aplicar la fragmentación híbrida dinámica y consultar los nuevos fragmentos a la tabla Fragmento (SELECT \* FROM FRAGMENTO WHERE ID\_TABLA = 188;) de la base de datos XAMANA; en la primera consulta que se realizó cuando se tenía la fragmentación híbrida estática (Figura 4.50), como se observa en la Figura 4.51 el primer fragmento se conservó, es decir, solo se actualizó el sitio, mientras que, el resto se eliminaron y se insertaron los nuevos fragmentos formados en el nuevo sitio y con su valor de desempeño.

id_fragmento	nombre	id_tabla	id_sitio	noperaciones	ndesempenio	isCBIR
1054	multimedia_records_1	188	6	100.000	204610.812	0
1055	multimedia_records_2	188	6	100.000	107764.219	0
1056	multimedia_records_3	188	6	100.000	142649.484	0
1057	multimedia_records_4	188	6	100.000	91449.977	0

Figura 4.50 Consulta sobre la base de datos XAMANA, fragmentación híbrida estática

id_fragmento	nombre	id_tabla	id_sitio	noperaciones	ndesempenio	isCBIR
1054	multimedia_records_1	188	5	100.000	204610.812	0
1058	multimedia_records_2_1	188	5	100.000	313992.844	0
1059	multimedia_records_2_2	188	5	100.000	0.000	0
1060	multimedia_records_3_1	188	5	100.000	569692.562	0
1061	multimedia_records_3_2	188	5	100.000	0.000	0
1062	multimedia_records_4_1	188	5	100.000	119881.961	0
1063	multimedia_records_4_2	188	5	100.000	119616.641	0

Figura 4.51 Consulta sobre la base de datos XAMANA, fragmentación híbrida dinámica

La Tabla 4.7 presenta el nuevo esquema de fragmentación híbrida dinámica del primer caso. En un inicio, el esquema de fragmentación híbrida estática se conformaba de 4 fragmentos. El fragmento *multimedia\_records\_1* no se refragmentó, mientras que, *multimedia\_records\_2*, *multimedia\_records\_3* y *multimedia\_records\_4* formaron, cada uno, dos fragmentos nuevos con diferentes atributos y número de tuplas, estas se describen a continuación.

Tabla 4.7 Fragmentos formados en la fragmentación híbrida dinámica, caso 1

Nombre del fragmento	Predicado	Atributos	Nro. de tuplas	Sitio origen	Sitio destino	Costo H-V	Costo V-H
<i>multimedia_records_1</i>	{"validado":{"\$eq":null}}	<i>descripción</i> <i>tipo</i> <i>validado</i> <i>_id</i> <i>nombre</i>	1843	2	1	1,323,249.76	1,323,249.76
<i>multimedia_records_2_1</i>	{"\$and":[{"_id":{"\$ne":"IMG1631"}}]}	<i>imagen</i> <i>descriptor</i> <i>_id</i>	1842	2	1	157,170.17	157,170.17
<i>multimedia_records_2_2</i>	{"\$and":[{"\$or":[{"_id":{"\$not":{"\$ne":"IMG1631"}}}],{"_id":{"\$ne":{"\$oid":"615e9812f0f0e30bc46131b5"}}}]}	<i>imagen</i> <i>descriptor</i> <i>_id</i>	1	2	1		
<i>multimedia_records_3_1</i>	{"\$and":[{"tipo":{"\$ne":"building"}}]}	<i>descripción</i> <i>tipo</i> <i>validado</i> <i>_id</i> <i>nombre</i>	13	2	1	302,599.14	302,599.14
<i>multimedia_records_3_2</i>	{"\$and":[{"\$or":[{"tipo":{"\$not":{"\$ne":"building"}}}],{"tipo":{"\$ne":"event"}}}]}	<i>descripción</i> <i>tipo</i> <i>validado</i> <i>_id</i> <i>nombre</i>	14	2	1		
<i>multimedia_records_4_1</i>	{"\$and":[{"_id":{"\$ne":"IMG841"}}]}	<i>imagen</i> <i>descriptor</i>					

Nombre del fragmento	Predicado	Atributos	Nro. de tuplas	Sitio origen	Sitio destino	Costo H-V	Costo V-H
		<i>_id</i>	26	2	1	59,942.49	59,942.49
<i>multimedia_records_4_2</i>	{ "\$and": [ { "\$or": [ { "_id": { "\$not": { "\$ne": "IMG841" } } } ] }, { "_id": { "\$ne": "IMG1924" } } ] }	<i>imagen descriptor _id</i>	1	2	1		

Los fragmentos que se encontraban en el sitio 2 (fragmentación híbrida estática) se eliminaron junto con la colección HITO, como se muestra en la Figura 4.52, esto es porque los nuevos fragmentos se asignaron al sitio 1.

La Figura 4.53 presenta la asignación y aplicación de cada uno de los fragmentos resultantes de la fragmentación híbrida dinámica, siendo un total de 7 fragmentos en el sitio 1.

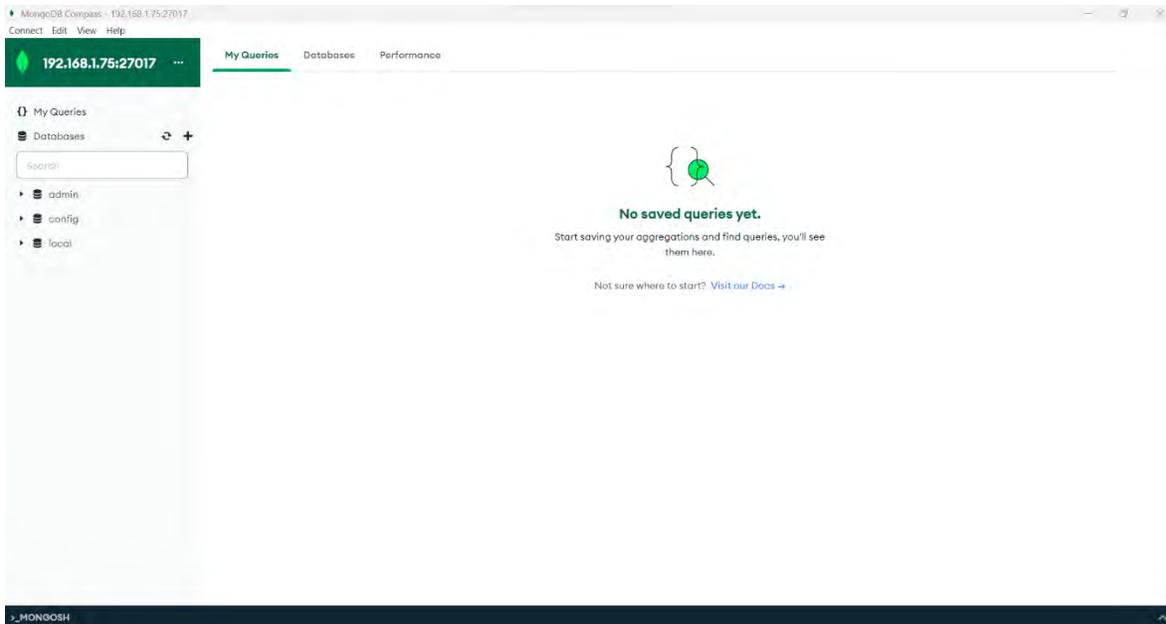


Figura 4.52 Eliminación del origen de los fragmentos primer caso, sitio 2

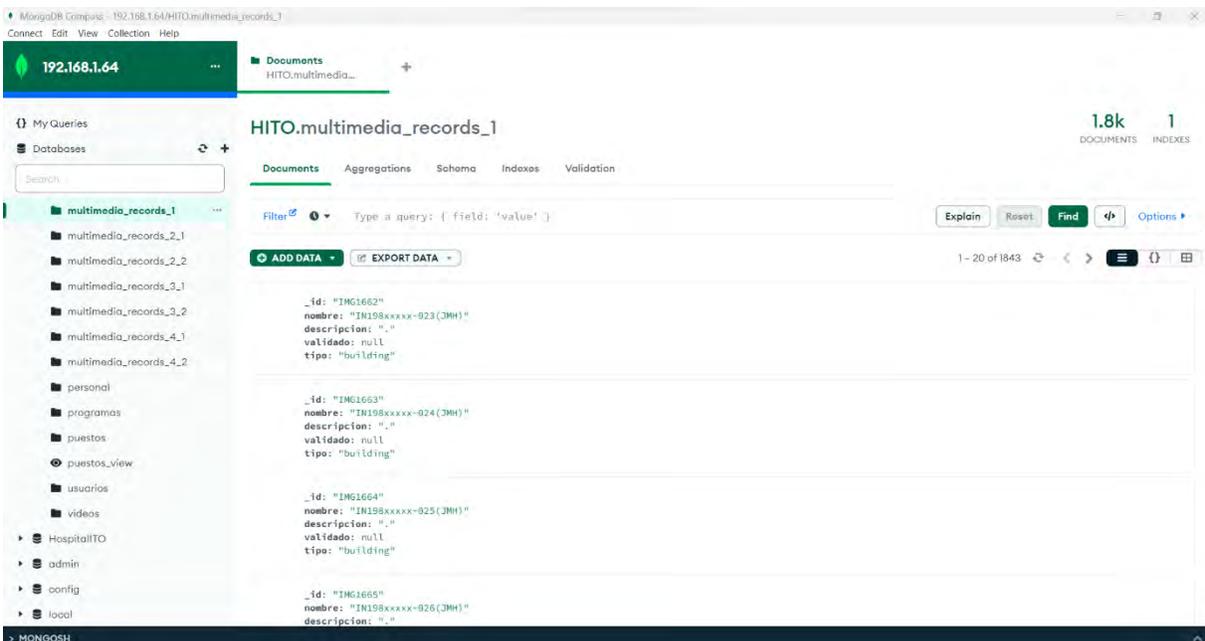


Figura 4.53 Asignación de los fragmentos híbridos en el sitio local, primer caso

En la Figura 4.54 se aprecia que, después de terminar de aplicar la fragmentación híbrida dinámica, *multimedia\_records\_1* se reasignó de forma correcta teniendo el mismo número de tuplas y atributos que en la Figura 4.31.

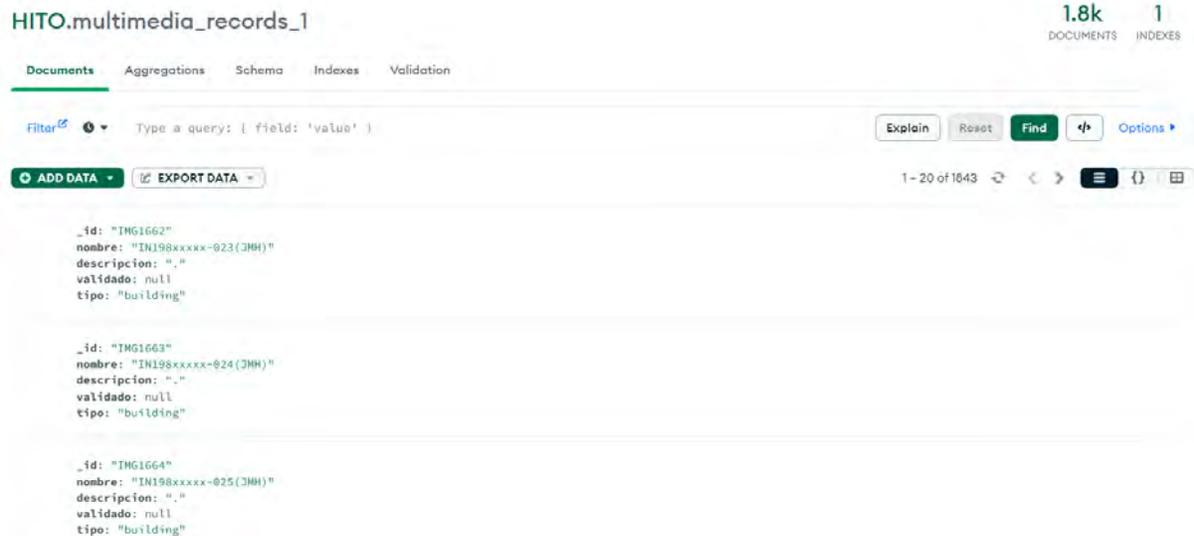


Figura 4.54 Fragmento “*multimedia\_records\_1*”, fragmentación híbrida dinámica, primer caso

Anteriormente, *multimedia\_records\_2* en la fragmentación híbrida estática tenía los atributos: *\_id*, *imagen* y *descriptor*, con 1843 tuplas. Se observa en las Figuras 4.55 y 4.56 que los fragmentos derivados de *multimedia\_records\_2\_1* contienen las tuplas adecuadas (1842 y 1) de los predicados que los definen, conservando los mismos atributos.

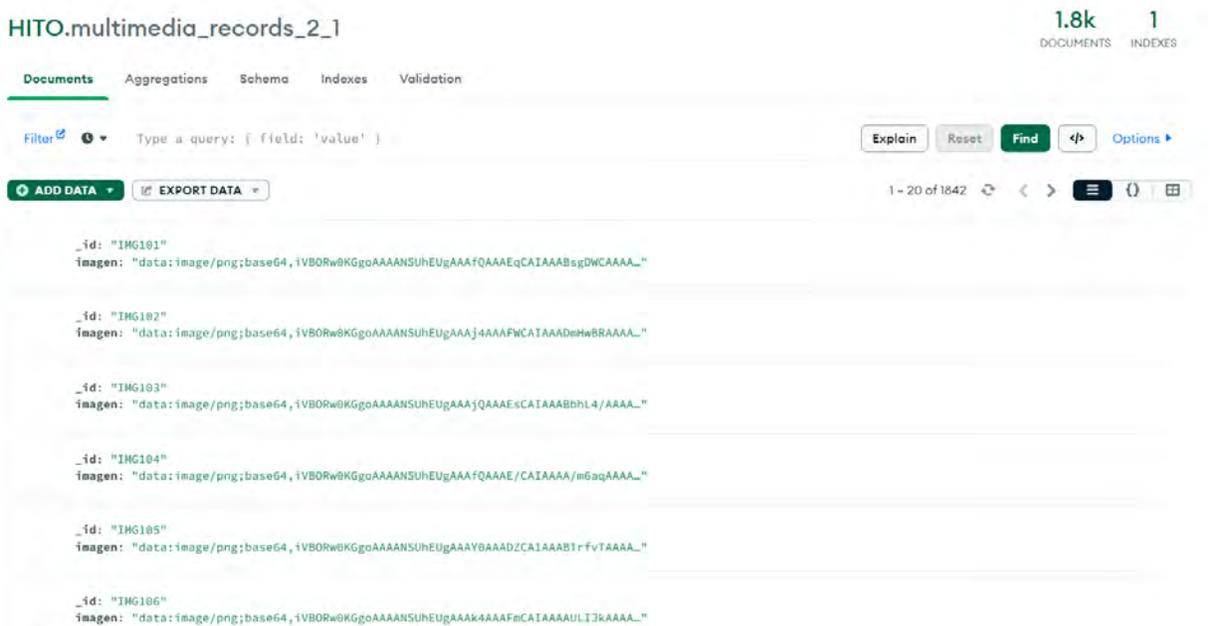


Figura 4.55 Fragmento “*multimedia\_records\_2\_1*”, fragmentación híbrida dinámica, primer caso



Figura 4.56 Fragmento “*multimedia\_records\_2\_2*”, fragmentación híbrida dinámica, primer caso

En las Figuras 4.57 y 4.58 se presentan los fragmentos obtenidos de *multimedia\_records\_3*. Ambos fragmentos poseen las tuplas correspondientes a cada uno de sus predicados  $\{ "\$and": [ \{ "tipo": \{ "\$ne": "building" \} \} ] \}$  y  $\{ "\$and": [ \{ "\$or": [ \{ "tipo": \{ "\$not": \{ "\$ne": "building" \} \} \} ] \}$ ,  $\{ "tipo": \{ "\$ne": "event" \} \} ] \}$ . Si se suman las tuplas de ambos fragmentos, da como resultado 27 tuplas que son las que tenía el fragmento estático (Figura 4.33).

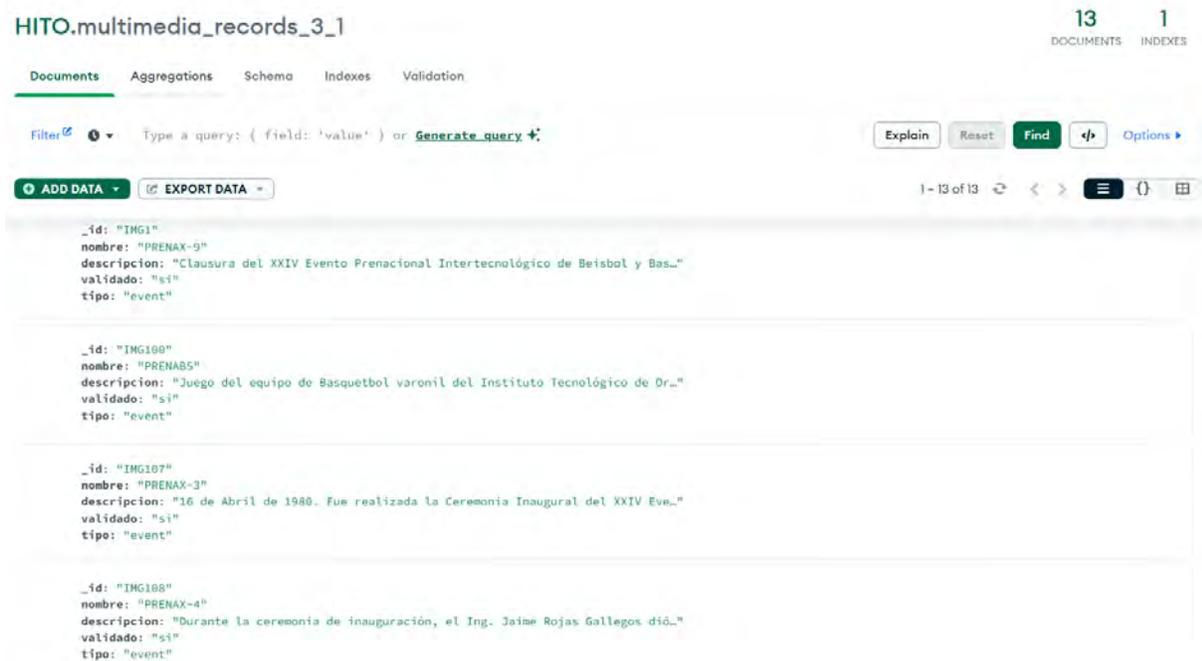


Figura 4.57 Fragmento “*multimedia\_records\_3\_1*”, fragmentación híbrida dinámica, primer caso

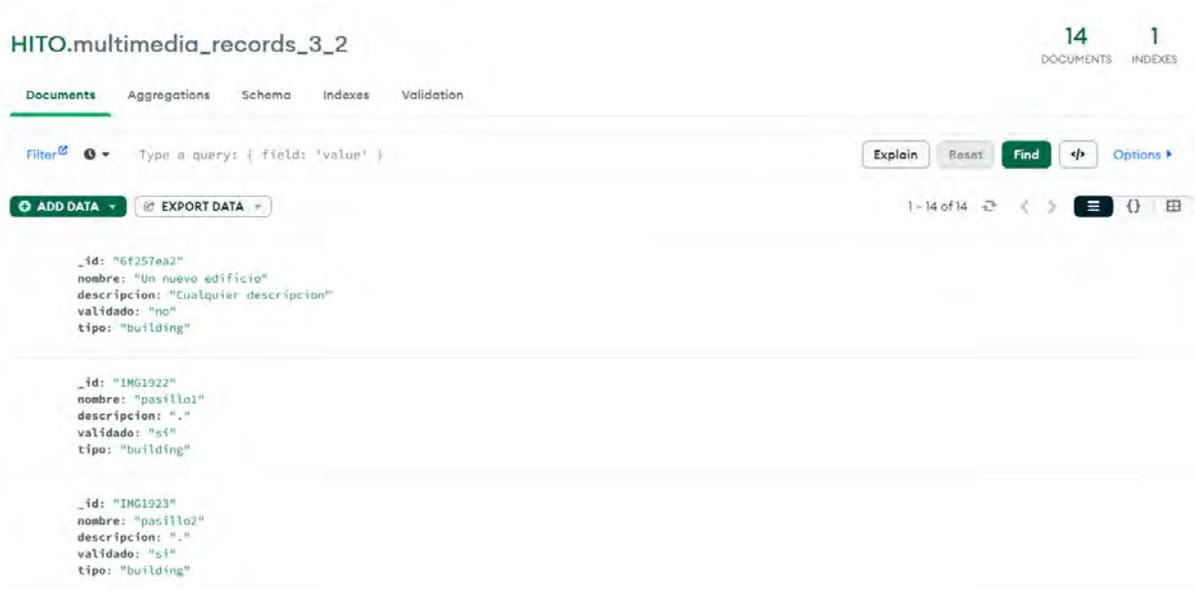


Figura 4.58 Fragmento “*multimedia\_records\_3\_2*”, fragmentación híbrida dinámica, primer caso

Finalmente, los últimos fragmentos de la partición híbrida dinámica son *multimedia\_records\_4\_1* y *multimedia\_records\_4\_2*, que se obtuvieron de *multimedia\_records\_4* (fragmentación híbrida estática). Son 26 documentos en *multimedia\_records\_4\_1* (Figura 4.59) donde el identificador no es igual a “IMG841” (`{"$and":[{"_id":{"$ne":"IMG841"}}]}`), mientras que, 1 documento satisface el predicado `{"$and":[{"$or":[{"_id":{"$not":{"$ne":"IMG841"}}}],{"_id":{"$ne":"IMG1924"}}]}` y se encuentra en *multimedia\_records\_4\_2* (Figura 4.60).

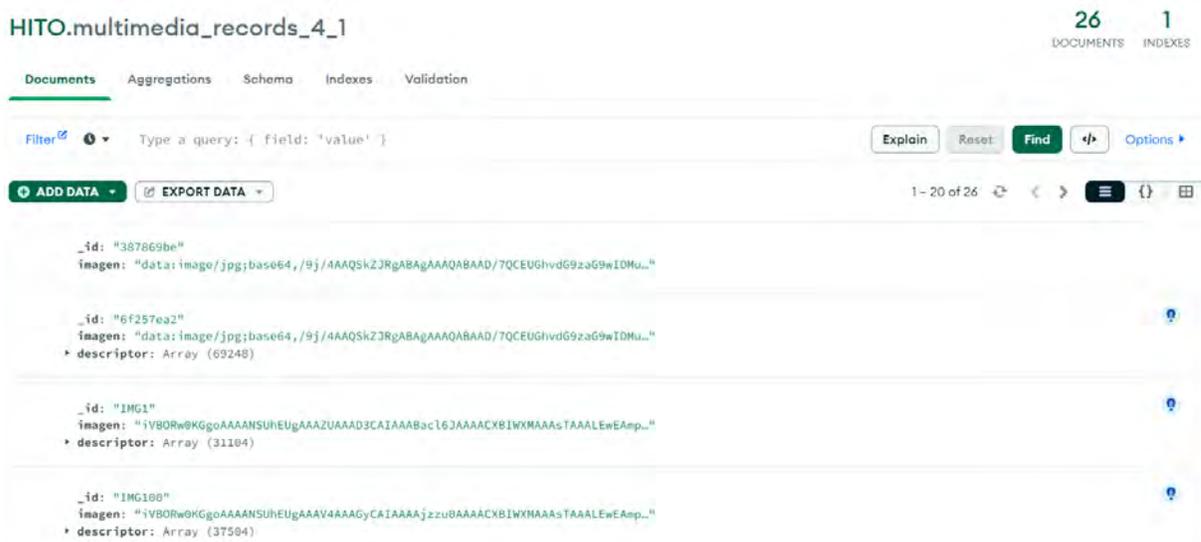


Figura 4.59 Fragmento “*multimedia\_records\_4\_1*”, fragmentación híbrida dinámica, primer caso



Figura 4.60 Fragmento “*multimedia\_records\_4\_2*”, fragmentación híbrida dinámica, primer caso

### 4.4.2 Fragmentación híbrida dinámica, segundo caso de estudio

A continuación, se presentan los resultados obtenidos en la fragmentación híbrida dinámica contemplando el esquema de fragmentación híbrida estática presentado en la Figura 4.24 (Resultado final de la fragmentación híbrida estática). La diferencia entre el primer caso y este, radica en que no todos los fragmentos alcanzaron el umbral de operaciones (*multimedia\_records\_4*).

Se realizan nuevas operaciones sobre los fragmentos híbridos. En la Figura 4.61 se presenta la nueva carga de trabajo para realizar la fragmentación híbrida dinámica (caso 2), esta igual se conforma de 4 archivos *logs*, los tres primeros que corresponden a los fragmentos: *multimedia\_records\_1*, *multimedia\_records\_2* y *multimedia\_records\_3* contienen 100 operaciones, siendo un total de 300 operaciones, mientras que en *multimeida\_records\_4*, solo se realizó una operación de consulta (Figura 4.62)

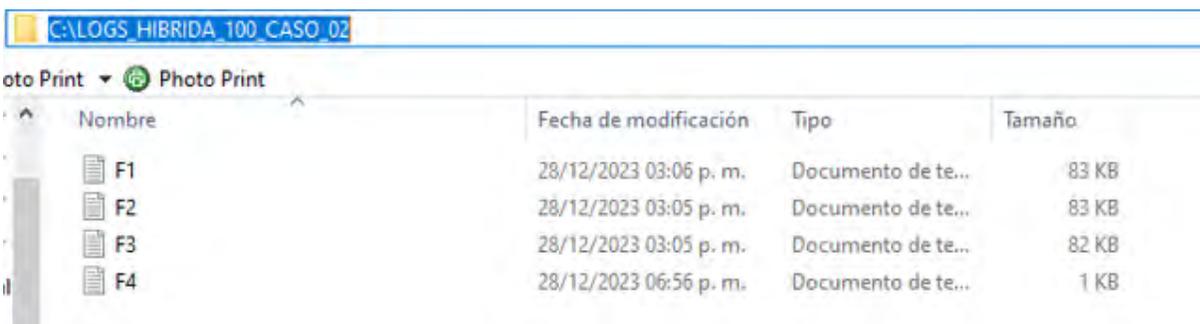


Figura 4.61 Carga de trabajo para la fragmentación híbrida dinámica, sitio remoto, segundo caso

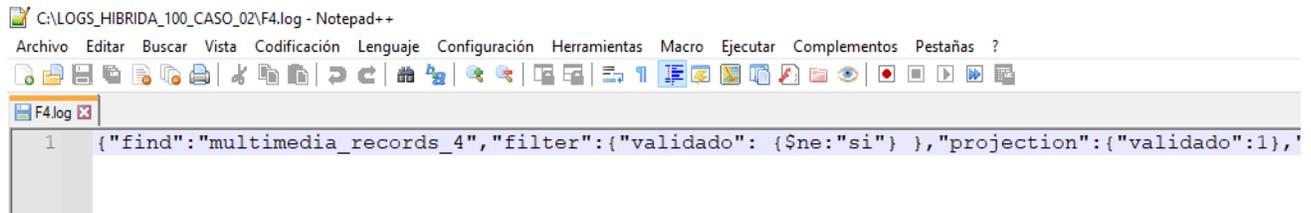


Figura 4.62 Operación de “multimedia\_records\_4”, segundo caso

En la Tabla 4.8 se detallan las operaciones por cada fragmento híbrido, por lo cual, el vigilante-fragmentador analizará 301 operaciones.

Tabla 4.8 Operaciones de la carga de trabajo para la fragmentación híbrida dinámica, segundo caso

Nombre del archivo	<i>Insert</i>		<i>Find</i>		<i>Update</i>		<i>Delete</i>	
F1_100	24	Sitio 1: 13	23	Sitio 1: 12	30	Sitio 1: 17	23	Sitio 1: 8
		Sitio 2: 11		Sitio 2: 11		Sitio 2: 13		Sitio 2: 15
F2_100	22	Sitio 1: 12	24	Sitio 1: 10	30	Sitio 1: 16	24	Sitio 1: 12
		Sitio 2: 10		Sitio 2: 14		Sitio 2: 14		Sitio 2: 12
F3_100	33	Sitio 1: 13	24	Sitio 1: 16	21	Sitio 1: 9	22	Sitio 1: 12
		Sitio 2: 20		Sitio 2: 8		Sitio 2: 12		Sitio 2: 10
F4_100	0	Sitio 1: 0	1	Sitio 1: 1	0	Sitio 1: 0	0	Sitio 1: 0
		Sitio 2: 0		Sitio 2: 0		Sitio 2: 0		Sitio 2: 0

De la misma manera que el primer caso de estudio, se requiere que la carga de trabajo se ubique donde se formaron los fragmentos híbridos, para que el vigilante en ese sitio lea las operaciones y analice qué fragmentos alcanzaron el umbral de operaciones, posteriormente calcule los costos de cada fragmento y detecte qué fragmentos alcanzaron el umbral de desempeño.

Con base en lo anterior, a continuación, se reporta:

- Vigilante-fragmentador desde el sitio 2
- Cuáles fueron los fragmentos que alcanzaron los umbrales
- El análisis de dos operaciones del archivo *log* del primer fragmento, vigilante en sitio remoto
- Costos de los fragmentos
- Esquema vertical y horizontal antes de obtener el esquema híbrido
- Resultados del esquema de fragmentación híbrida
- Comparación del esquema estático con el dinámico

Para llevar a cabo la nueva evaluación, se ejecuta el vigilante-fragmentador en todos los sitios. A continuación, se presenta la ejecución del vigilante en el sitio 2, este regresa la información de la base de datos con la colección a fragmentar y los umbrales (Figura 4.63). Para el vigilante que se encuentra en el otro sitio (local), al no tener fragmentos no se analiza ninguna operación.

```
run:
Enter token obtained from the Web application to initiate the method (The token is only an integer)
190
Enter database log location: C:\LOGS
C:\LOGS_HIBRIDA_100_CASO_02

-----
Data found and information obtained
-----
Database name           HITO
Database IP            192.168.1.64
Fragmented table       multimedia_records
Operation threshold     2.0
Performance threshold  2.0
Total operation log     100.0
-----

Fragments sites finds
Fragment name: multimedia_records_1
Fragment name: multimedia_records_2
Fragment name: multimedia_records_3
Fragment name: multimedia_records_4
Fragments scheme found
```

Figura 4.63 Vigilante-fragmentador, sitio remoto, segundo caso

Se supervisaron que fragmentos alcanzaron el umbral de operaciones (Figura 4.64) y solo: *multimedia\_records\_1*, *multimedia\_records\_2* y *multimedia\_records\_3* cumplieron, mientras que, *multimedia\_records\_4* (Figura 4.65) no alcanzó el umbral y no se considera para el análisis de costos.

```
-----
Current operation information for hybrid fragmentation (first analysis):
-----
Fragment name:      multimedia_records_1
Achieve Operation:      2.0
Number of Operations:      100
-----
-----
Current operation information for hybrid fragmentation (first analysis):
-----
Fragment name:      multimedia_records_2
Achieve Operation:      2.0
Number of Operations:      100
-----
-----
Current operation information for hybrid fragmentation (first analysis):
-----
Fragment name:      multimedia_records_3
Achieve Operation:      2.0
Number of Operations:      100
-----
```

Figura 4.64 Fragmentos que alcanzaron el umbral de operaciones, segundo caso

```
-----
Current operation information for hybrid fragmentation (first analysis):
-----
Fragment name:      multimedia_records_4
Achieve Operation:      2.0
Number of Operations:      1
-----
-----
Remove fragment: multimedia_records_4
```

Figura 4.65 Fragmento que no alcanzó el umbral de operaciones, segundo caso

Se realiza el segundo paso (analizar el costo del esquema actual), por lo tanto, se procede con el análisis de operaciones. Se seleccionaron dos operaciones del archivo *log* del primer fragmento “*multimedia\_records\_1*”, como se muestra en la Figura 4.6, donde la operación 13 es una eliminación y la 98 es una inserción. En la Tabla 4.9 se explica detalladamente el cálculo de las operaciones 13 y 98 seleccionadas del archivo *log* del primer fragmento y que corresponden a las mostradas en la Figuras 4.66; son más operaciones las que se analizan (400 operaciones), pero solo se presentan dos.

```

Number of operation: 13
{"t":{"$date":"2023-10-14T14:43:21.284000Z"},"s":"I", "c":"COMMAND", "id":7761,"ctx":"conn1","msg":"Slow query","attr":{"typ
Count of tuples: {"_id":{"$ne":{"$oid":"615e99b0f0f0e30bc4613379"}}}
Number of tuples: 1842.0

Count of tuples: {"_id":{"$ne":{"$oid":"615e99b0f0f0e30bc4613379"}}}
Number of tuples: 1842.0

Count of tuples: {"_id":{"$ne":{"$oid":"615e99b0f0f0e30bc4613379"}}}
Number of tuples: 27.0

Count of tuples: {"_id":{"$ne":{"$oid":"615e99b0f0f0e30bc4613379"}}}
Number of tuples: 27.0

VAO: 4
CR: 4

Cost of vertical operation: 0.16
Cost of horizontal operation: 14952.0
Cost of hybrid operation: 7476.08
The cost of operation is allocated to the fragment: multimedia_records_1
The cost of operation is allocated to the fragment: multimedia_records_2
The cost of operation is allocated to the fragment: multimedia_records_3
The cost of operation is allocated to the fragment: multimedia_records_4

Number of operation: 98
{"t":{"$date":"2023-10-14T14:43:44.996000Z"},"s":"I", "c":"COMMAND", "id":3474,"ctx":"conn2","msg":"Slow query","attr":{"typ
Count of tuples: {"_id":{"$oid":"615e9692f0f0e30bc4613089"}}}
Number of tuples: 1.0

Count of tuples: {"_id":{"$oid":"615e9692f0f0e30bc4613089"}}}
Number of tuples: 1.0

Count of tuples: {"_id":{"$oid":"615e9692f0f0e30bc4613089"}}}
Number of tuples: 0.0

Count of tuples: {"_id":{"$oid":"615e9692f0f0e30bc4613089"}}}
Number of tuples: 0.0

VAO: 2
CR: 2

Cost of vertical operation: 0.04
Cost of horizontal operation: 4.0
Cost of hybrid operation: 2.02
The cost of operation is allocated to the fragment: multimedia_records_1
The cost of operation is allocated to the fragment: multimedia_records_2

```

Figura 4.66 Análisis de operaciones del vigilante-fragmentador, sitio remoto, segundo caso

Tabla 4.9 Análisis de costo de operaciones en *multimedia\_records\_1*, fragmentación híbrida dinámica, segundo caso

<b>Criterio</b>	<b>Resultado</b>
<b>Nro. de operación: 13</b>	db.multimedia_records.deleteMany({"_id":{"\$ne:ObjectId("615e99b0f0f0e30bc4613379")}})
<b>Tipo y valor de la operación:</b>	Eliminación ( <i>delete</i> ) = <b>2</b>
<b>Predicado de la operación:</b>	{"_id":{"\$ne:ObjectId("615e99b0f0f0e30bc4613379")}}
<b>VAO:</b>	2 (valor remoto) * 2 del valor de operación = <b>4</b>
<b>CR:</b>	<b>4</b> fragmentos que se requieren para la fragmentación
<b>Número de tuplas por predicado:</b>	En 2 fragmentos se encontraron 1842 tuplas que contienen el predicado = 3,684 tuplas. En 2 fragmentos se encontraron 27 tuplas que contienen el predicado = 54 tuplas. <b>Total de tuplas: 3,684 + 54 = 3,738</b>
<b>Costo horizontal:</b>	$Hor = tMax * VAO * VR$ $Hor = 3,738 * 4 * 1 = \mathbf{14,952}$
<b>Costo vertical:</b>	$Ver = VAO * CR * VR * \sum_{i=1}^N Ta_i$ $Ver = 4 * 4 * 1 * 0.01 = \mathbf{0.16}$
<b>Costo híbrido:</b>	$Hib = \frac{(Hor+Ver)}{2}$ $Hib = (14,952 + 0.16) / 2$ $Hib = \mathbf{7,476.08}$
<b>Nro. de operación: 98</b>	db.multimedia_records.insertOne({"_id":ObjectId("615e9692f0f0e30bc4613089")})
<b>Tipo y valor de la operación:</b>	Inserción ( <i>insert</i> ) = <b>2</b>

Criterio	Resultado
<b>Predicado de la operación:</b>	{"_id":ObjectId("615e9692f0f0e30bc4613089")}
<b>VAO:</b>	1 (valor local) * 2 del valor de operación = <b>2</b>
<b>CR:</b>	2 fragmentos que se requieren para la fragmentación
<b>Número de tuplas por predicado:</b>	2 fragmentos se encontraron con 1 tuplas que contienen el predicado = 2. <b>Total de tuplas: 2</b>
<b>Costo horizontal:</b>	$Hor = tMax * VAO * VR$ $Hor = 2 * 2 * 1 = 4$
<b>Costo vertical:</b>	$Ver = VAO * CR * VR * \sum_{i=1}^N Ta_i$ $Ver = 2 * 2 * 1 * 0.01 = 0.04$
<b>Costo híbrido:</b>	$Hib = \frac{(Hor+Ver)}{2}$ $Hib = (4 + 0.04) / 2$ $Hib = 2.02$

En la Figura 4.67 se muestra el costo del fragmento *multimedia\_records\_1* después de realizar el análisis de costos y en la Tabla 10 del resto de los fragmentos, pero ahora sobre el segundo caso.

```
-----
Current cost information for hybrid fragmentation (second analysis):
-----
Cost in multimedia_records_1:    609762.2650000016
Performance threshold:          2.0
Initial performance:            204610.812
Achieve performance:           4092.21624
-----

This fragment is not CBIR

-----CREATE HYBRID SCHEME-----
```

Figura 4.67 Información del costo del fragmento “*multimedia\_records\_1*”, segundo caso

Tabla 4.10 Comparación de los costos de los fragmentos híbridos, segundo caso

Nombre del fragmento	Costo	Umbral de desempeño	Rendimiento inicial	Rendimiento a alcanzar
<i>multimedia_records_2</i>	522,971.855	2	107,764.219	2,155.284
<i>multimedia_records_3</i>	559,202.520		142,649.484	2,852.989

Los tres fragmentos alcanzaron el umbral de desempeño, por consiguiente, se realiza la lectura del archivo *log* para crear el esquema vertical y horizontal para obtener el esquema de fragmentación híbrida dinámica. La Figura 4.68 muestra el resultado final del esquema de partición vertical y horizontal contemplando a *multimedia\_records\_1*.

```

Vertical scheme
Fragment name: multimedia_records_1
Attributes: descripcion
Attributes: tipo
Attributes: validado
Attributes: nombre
Attributes: _id

Horizontal scheme
Fragment name: multimedia_records_1
Predicate: {"_id":{"$ne":"IMG1909"}}
Overlap: {"$and":[{"_id":{"$ne":"IMG1909"}}]}

Horizontal scheme
Fragment name: multimedia_records_2
Predicate: {"_id":{"$ne":"IMG449"}}
Overlap: {"$and":[{"$or":[{"_id":{"$not":{"$ne":"IMG1909"}}}],{"_id":{"$ne":"IMG449"}}]}
    
```

Figura 4.68 Esquema vertical y horizontal dinámico considerando “*multimedia\_records\_1*”, segundo caso

La Tabla 4.11 compara el resto de los fragmentos verticales y horizontales formados en la partición dinámica, antes de crear el esquema híbrido.

Tabla 4.11 Comparación de los fragmentos verticales y horizontales en la partición dinámica, segundo caso

Nro. de fragmento	Nombre del fragmento	Esquema Vertical	Esquema Horizontal
2	<i>multimedia_records_1</i>	<i>imagen descriptor _id</i>	<b>Predicado:</b> {"_id":{"\$ne":{"\$oid":"615e997af0f0e30bc46132ad"}}} <b>Traslape:</b> {"\$and":[{"_id":{"\$ne":{"\$oid":"615e997af0f0e30bc46132ad"}}}]}
	<i>multimedia_records_2</i>	-----	<b>Predicado:</b> {"_id":{"\$ne":"IMG768"}} <b>Traslape:</b> {"\$and":[{"\$or":[{"_id":{"\$not":{"\$ne":{"\$oid":"615e997af0f0e30bc46132ad"}}}}]}, {"_id":{"\$ne":"IMG768"}}]}
3	<i>multimedia_records_1</i>	<i>descripción tipo validado _id nombre</i>	<b>Predicado:</b> {"_id":{"\$ne":"IMG1930"}} <b>Traslape:</b> {"\$and":[{"_id":{"\$ne":"IMG1930"}}]}
	<i>multimedia_records_2</i>	-----	<b>Predicado:</b> {"_id":{"\$ne":"IMG841"}} <b>Traslape:</b> {"\$and":[{"\$or":[{"_id":{"\$not":{"\$ne":"IMG1930"}}}], {"_id":{"\$ne":"IMG841"}}]}]}

Se obtiene el nuevo esquema de partición híbrida dinámica por cada fragmento antes de refragmentar y asignar. La Figura 4.69 muestra la información de los dos nuevos fragmentos

derivados de *multimedia\_records\_1*. En este caso se formaron dos fragmentos, *multimedia\_records\_1\_1* y *multimedia\_records\_1\_2*.

```

-----
Hybrid Scheme
-----
Total tuples of the original fragment: 1843.0
Total tuples of the new fragment: 1842
Fragment name: multimedia_records_1_1
Overlap: {"$and":[{"_id":{"$ne":"IMG1909"}}]}
Attribute: descripcion
Attribute: tipo
Attribute: validado
Attribute: nombre
Attribute: _id

Total tuples of the original fragment: 1843.0
Total tuples of the new fragment: 1
Fragment name: multimedia_records_1_2
Overlap: {"$and":[{"_id":{"$not":{"$ne":"IMG1909"}}}],{"_id":{"$ne":"IMG449"}}]}
Attribute: descripcion
Attribute: tipo
Attribute: validado
Attribute: nombre
Attribute: id
-----

Both schemes have equals cost but this will be applied vertical-horizontal scheme

```

Figura 4.69 Esquema de la fragmentación híbrida dinámica “*multimedia\_records\_1\_1*” y “*multimedia\_records\_2\_1*”, segundo caso

Finalmente, el nuevo esquema de fragmentación híbrida se asigna y aplica al sitio correspondiente (sitio 1), se actualiza la información en la base de datos XAMANA (Figura 4.70) y el vigilante del sitio 1 (local) detecta que se ha realizado una refragmentación, actualizando su información, como se observa en la Figura 4.71.

```

-----
Apply new Scheme Hybrid
-----
Fragment name: multimedia_records_1_1
Destination: 192.168.1.64
insert into Fragmento(nombre, id_tabla, id_sitio, noperaciones, ndesempenio, isCBIR) value ('multimedia_records_1_1')
Fragment name: multimedia_records_1_2
Destination: 192.168.1.64
insert into Fragmento(nombre, id_tabla, id_sitio, noperaciones, ndesempenio, isCBIR) value ('multimedia_records_1_2')
Deleting origin.....
-----

```

Figura 4.70 Asignación de los nuevos fragmentos híbridos “*multimedia\_records\_1\_1*” y “*multimedia\_records\_1\_2*”, segundo caso

```
Re-fragment flag: true
Accepting connections over socket
Re-fragment flag: true
Accepting connections over socket

Fragments sites finds
Fragment name: multimedia_records_1_1
Fragment name: multimedia_records_1_2
Fragments remote sites finds
Fragment name: multimedia_records_2
Fragment name: multimedia_records_3
Fragment name: multimedia_records_4
```

Figura 4.71 Actualización de vigilante, sitio 1, segundo caso

El resultado obtenido en la fragmentación híbrida dinámica del segundo caso, son 6 fragmentos derivados de la segunda carga de trabajo; en la Tabla 4.13 se detallada cada fragmento formado.

Tabla 4.12 Fragmentos formados en la fragmentación híbrida dinámica, caso 2

Nombre del fragmento	Predicado	Atributos	Nro. de tuplas	Sitio origen	Sitio destino	Costo H-V	Costo V-H
<i>multimedia_records_1_1</i>	<code>{"_id":{"ne":"IMG1909"}}</code>	<i>descripción</i> <i>tipo</i> <i>validado</i> <i>_id</i> <i>nombre</i>	1842	2	1	300,711.34	300,711.34
<i>multimedia_records_1_2</i>	<code>{"_or":{"_id":{"not":{"ne":"IMG1909"}}, {"_id":{"ne":"IMG449"}}}</code>	<i>descripción</i> <i>tipo</i> <i>validado</i> <i>_id</i> <i>nombre</i>	1	2	1		
<i>multimedia_records_2_1</i>	<code>{"_id":{"ne":{"soid":"615e997af0f0e30bc46132ad"}}</code>	<i>imagen</i> <i>descriptor</i> <i>id</i>	1842	2	1	304,272.85	304,272.85
<i>multimedia_records_2_2</i>	<code>{"_or":{"_id":{"not":{"ne":{"soid":"615e997af0f0e30bc46132ad"}}, {"_id":{"ne":"IMG768"}}}</code>	<i>imagen</i> <i>descriptor</i> <i>_id</i>	1	2	1		
<i>multimedia_records_3_1</i>	<code>{"_id":{"ne":"IMG1930"}}</code>	<i>descripción</i> <i>tipo</i> <i>validado</i> <i>_id</i> <i>nombre</i>	26	2	1	78,047.62	78,047.62
<i>multimedia_records_3_2</i>	<code>{"_or":{"_id":{"not":{"ne":"IMG1930"}}, {"_id":{"ne":"IMG841"}}}</code>	<i>descripción</i> <i>tipo</i> <i>validado</i> <i>_id</i> <i>nombre</i>	1	2	1		

A continuación, se muestran los resultados obtenidos en los fragmentos híbridos del segundo caso. En el sitio 2 se conservó *multimedia\_records\_4* con sus atributos y documentos originales (*\_id*, *imagen* y *descriptor*, con 27 documentos) porque este fragmento no alcanzó el umbral de operaciones, tal como se muestra en la Figura 4.72. Por otra parte, los fragmentos que se refragmentaron se asignaron al sitio 1 como se observa en la Figura 4.73.

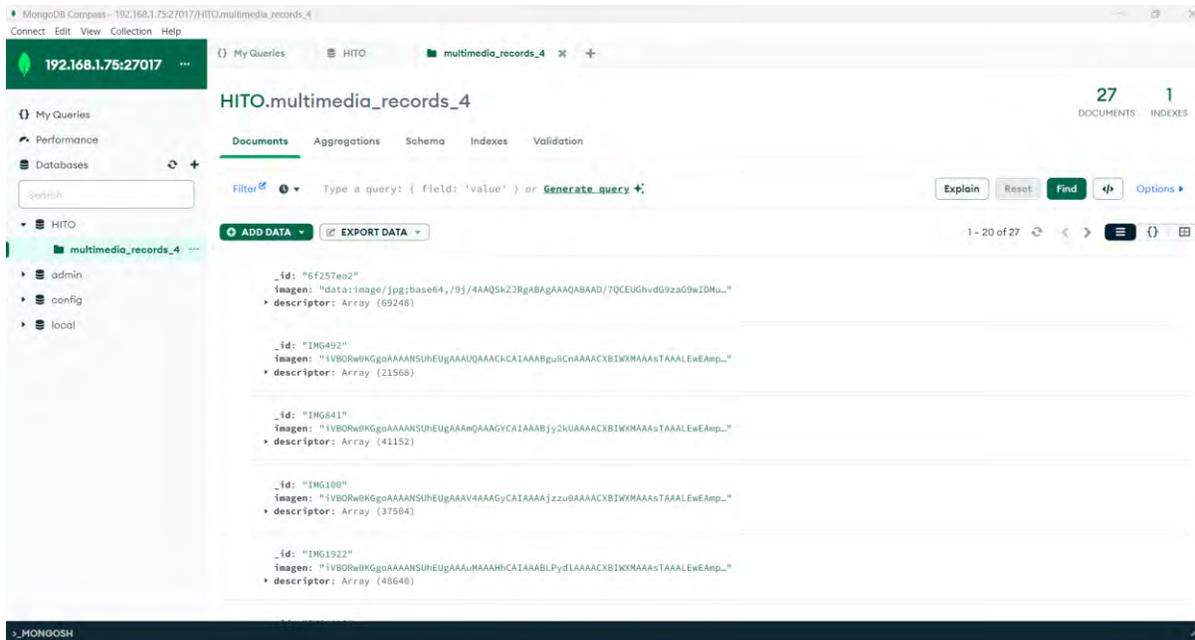


Figura 4.72 Sitio 2, con el fragmento “*multimedia\_records\_4*”, segundo caso

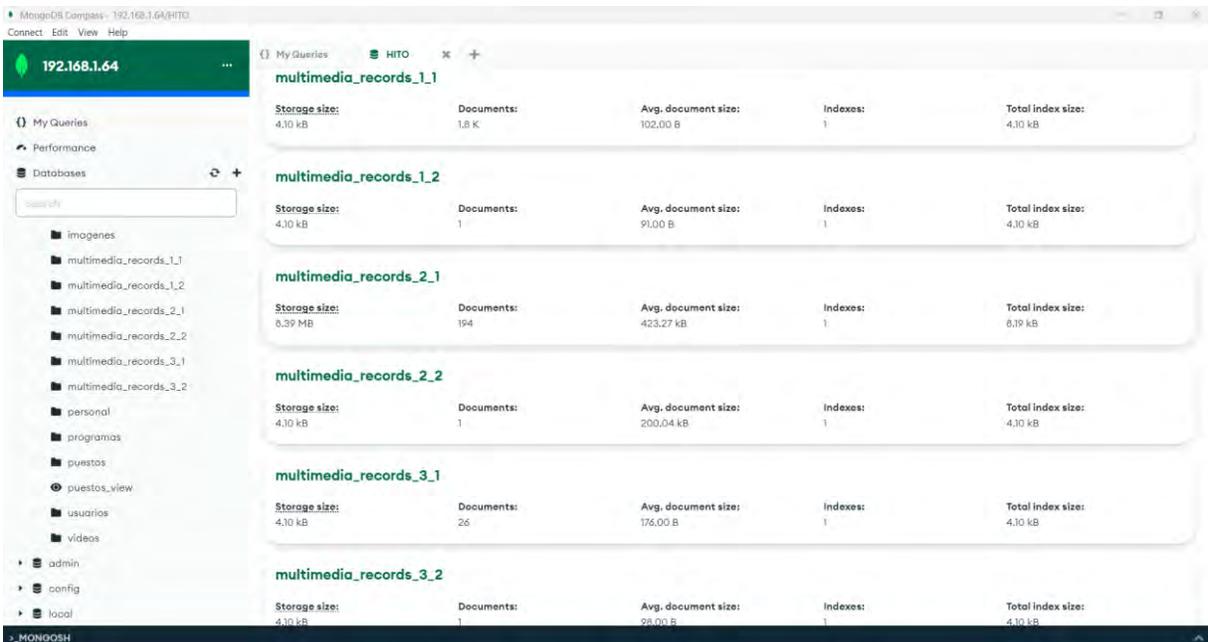


Figura 4.73 Asignación de los fragmentos híbridos en el sitio local, segundo caso

El total de documentos que conforman al fragmento *multimedia\_records\_1\_1* son 1842, con los atributos *\_id*, *nombre*, *descripción*, *validado* y *tipo* (Figura 4.74) y en *multimedia\_records\_1\_2*, hay un documento de acuerdo al predicado: `{ "$and": [ { $or: [ { "_id": { $not: { "$ne": "IMG1909" } } } ] }, { "_id": { "$ne": "IMG449" } } ] }`, con los mismos atributos del primer fragmento, tal como se presenta en la Figura 4.75.



Figura 4.74 Fragmento “*multimedia\_records\_1\_1*”, fragmentación híbrida dinámica, segundo caso

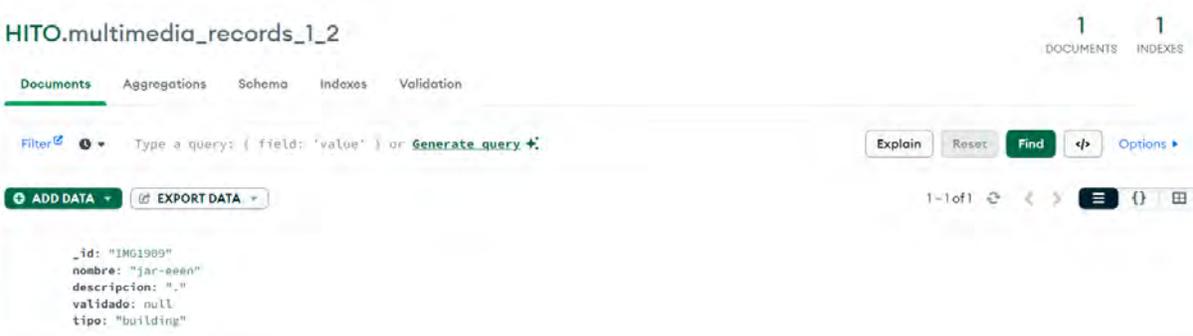


Figura 4.75 Fragmento “*multimedia\_records\_1\_2*”, fragmentación híbrida dinámica, segundo caso

En *multimedia\_records\_2\_1* existen 1842 documentos que cumplen con el predicado: `{"$and":[{"_id":{"$ne":{"$oid":"615e997af0f0e30bc46132ad"}}}]}`, con los atributos *id*, *imagen* y *descriptor* y se presenta en la Figura 4.76.

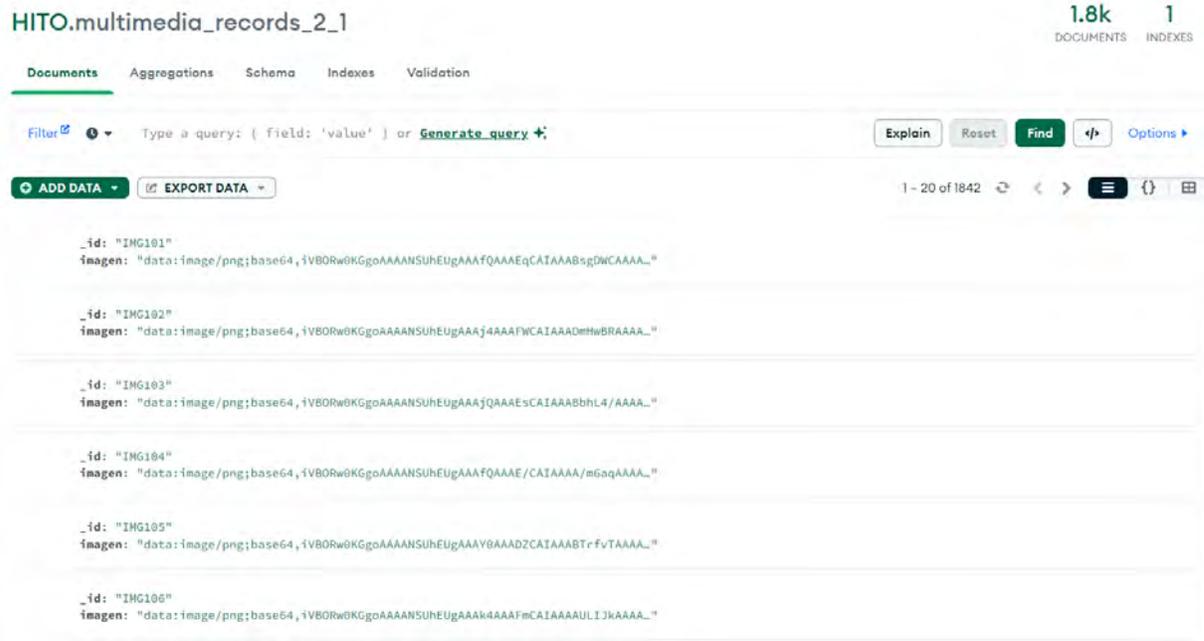


Figura 4.76 Fragmento “*multimedia\_records\_2\_1*”, fragmentación híbrida dinámica, segundo caso

Mientras que, *multimedia\_records\_2\_2* se conforma de un documento, donde el predicado es: `{"$and":[{"$or":[{"_id":{"$not":{"$ne":{"$oid":"615e997af0f0e30bc46132ad"}}}]}, {"_id":{"$ne":"IMG768"}}]}]}` (Figura 4.77)



Figura 4.77 Fragmento “*multimedia\_records\_2\_2*”, fragmentación híbrida dinámica, segundo caso

Por último, existen 26 documentos en *multimedia\_records\_3\_1* (Figura 4.78) donde el identificador no es igual a "IMG1930", con los atributos *\_id*, *nombre*, *descripción*, *validado* y *tipo* y solo en un documento el identificador no es igual a "IMG841" (Figura 4.79) y se encuentra en el fragmento "*multimedia\_records\_3\_2*".

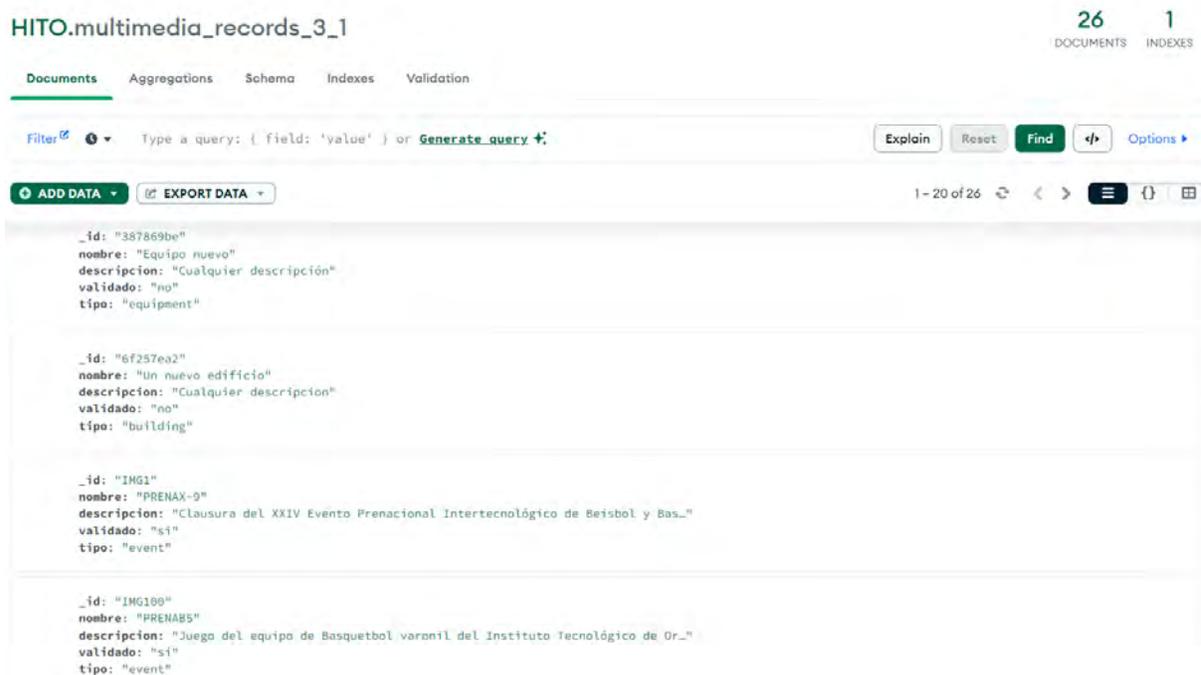


Figura 4.78 Fragmento "*multimedia\_records\_3\_1*", fragmentación híbrida dinámica, segundo caso

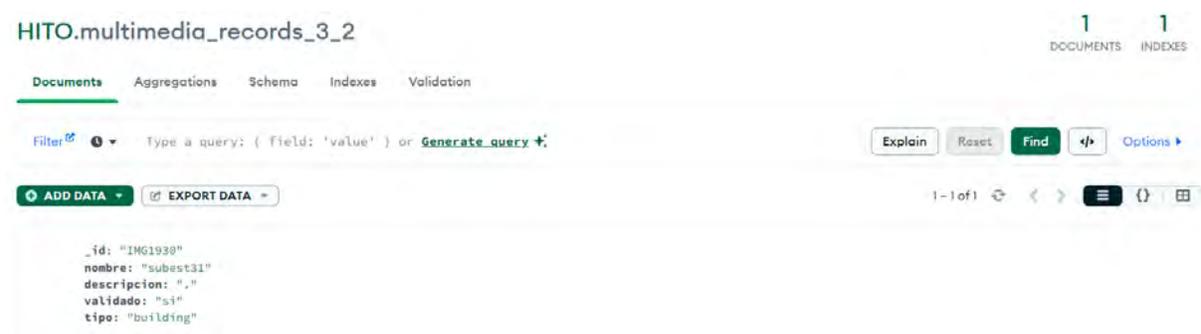


Figura 4.79 Fragmento "*multimedia\_records\_2\_2*", fragmentación híbrida dinámica, segundo caso

A continuación, se presentan tres tablas con el objetivo de realizar una comparación del resultado obtenido sin fragmentar, aplicando la fragmentación híbrida estática y, por último, la fragmentación híbrida dinámica contemplando el primer caso y el segundo.

Las Tablas 4.13 y 4.14 se encuentran conformadas de la siguiente manera: la primera fila muestra la colección *multimedia\_records* en HITO, con los siete atributos (*\_id*, *nombre*, *descripción*, *imagen*, *validado*, *descriptor* y *tipo*) y los 1870 documentos, posteriormente, las siguientes cuatro filas (en ambas tablas) presentan el resultado obtenido de la fragmentación híbrida estática, mientras que, el resto de las filas en (Tabla 4.14 y Tabla 4.15) corresponden a los resultados obtenidos en ambos casos de la fragmentación híbrida dinámica.

En un inicio, la colección *multimedia\_records* se encuentra completa en el sitio 1 (local), al momento de aplicar la fragmentación híbrida estática, se dividió la colección en 4 fragmentos que se asignaron al sitio 2 (remoto) con base en la primera carga (Tabla 4.1). Cuando se llevó a cabo la híbrida dinámica (primer y segundo caso), se refragmentó de acuerdo a la segunda carga de trabajo (Tabla 4.3 en el primer caso y Tabla 4.8 en el segundo caso) y el umbral de operaciones, obteniendo nuevos fragmentos.

Por otra parte, en la Tabla 4.15 se hace la comparación entre el esquema de fragmentación híbrida dinámica caso 1 y caso 2. Hay diferencias entre ambos casos, esto es porque, para el primer caso la carga de trabajo es de 400 operaciones (100 por cada fragmento) y todos lograron alcanzar el umbral de operaciones, obteniendo seis nuevos fragmentos asignados al sitio remoto; *multimedia\_records\_1* no se refragmentó porque en el nuevo esquema, resultó el mismo número de tuplas y atributos del fragmento original (1843), además, el vigilante lo asignó al sitio 1 porque fue donde más se utilizó. Mientras que, la carga de trabajo utilizada en el caso 2, fue diferente al anterior caso, además, se realizaron 301 operaciones, por lo tanto, los primeros fragmentos alcanzaron el umbral de operaciones (*multimedia\_records\_1*, *multimedia\_records\_2* y *multimedia\_records\_3*), el último (*multimedia\_records\_4*) no se consideró (no alcanzó el umbral de operaciones). Como resultado se formaron 6 nuevos fragmentos asignados al primer sitio.

Finalmente, los casos expuestos son diferentes y comprueban que para realizar la fragmentación híbrida dinámica es importante considerar el umbral de operaciones y desempeño.

Tabla 4.13 Comparación del esquema de fragmentación híbrida estática con fragmentación híbrida dinámica, primer caso

Nombre	Predicado	Atributos	Nro. de documentos	Sitio origen	Sitio destino	Costo H-V	Costo V-H
<b>Colección de la base de datos HITO a fragmentar</b>							
<i>multimedia_records</i>	N/A	<i>_id,</i> <i>nombre,</i> <i>descripción,</i> <i>imagen,</i> <i>validado,</i> <i>descriptor y</i> <i>tipo</i>	1870	1	N/A	N/A	N/A
<b>Fragmentación híbrida estática</b>							
<i>multimedia_records_1</i>	{"\$and":[{"validado":{"\$eq":null}}]}	<i>_id,</i> <i>nombre,</i> <i>descripción,</i> <i>validado y</i> <i>tipo</i>	1843	1	2	205,341.259	205,341.259
<i>multimedia_records_2</i>	{"\$and":[{"validado":{"\$eq":null}}]}	<i>_id, imagen</i> <i>y descriptor</i>	1843				
<i>multimedia_records_3</i>	{"\$and":[{"\$or":[{"validado":{"\$not":{"\$eq":null}}}],{"validado":{"\$ne":null}}]}	<i>_id,</i> <i>nombre,</i> <i>descripción,</i> <i>validado y</i> <i>tipo</i>	27				
<i>multimedia_records_4</i>	{"\$and":[{"\$or":[{"validado":{"\$not":{"\$eq":null}}}],{"validado":{"\$ne":null}}]}	<i>_id, imagen</i> <i>y descriptor</i>	27				
<b>Fragmentación híbrida dinámica, primer caso</b>							
<i>multimedia_records_1</i>	{"\$and":[{"validado":{"\$eq":null}}]}	<i>_id,</i> <i>nombre,</i> <i>descripción,</i> <i>validado y</i> <i>tipo</i>	1843			264,566.76	264,566.76

Nombre	Predicado	Atributos	Nro. de documentos	Sitio origen	Sitio destino	Costo H-V	Costo V-H
<i>multimedia_records_2_1</i>	{"Sand":[{"_id":{"\$ne:"IMG1631"}}]}	<i>_id, imagen y descriptor</i>	1842	2	1	157,170.17	157,170.17
<i>multimedia_records_2_2</i>	{"Sand":[{"\$or":[{"_id":{"\$not":{"\$ne":"IMG1631"}}}], [{"_id":{"\$ne":{"\$oid":"615e9812f0f0e30bc46131b5"}}]}]}	<i>_id, imagen y descriptor</i>	1				
<i>multimedia_records_3_1</i>	{"Sand":[{"tipo":{"\$ne:"building"}}]}	<i>_id, nombre, descripción, validado y tipo</i>	13			302,599.14	302,599.14
<i>multimedia_records_3_2</i>	{"Sand":[{"\$or":[{"tipo":{"\$not":{"\$ne":"building"}}}], [{"tipo":{"\$ne":"event"}}]}]}	<i>_id, nombre, descripción, validado y tipo</i>	14				
<i>multimedia_records_4_1</i>	{"Sand":[{"_id":{"\$ne:"IMG841"}}]}	<i>_id, imagen y descriptor</i>	26			59,942.49	59,942.49
<i>multimedia_records_4_2</i>	{"Sand":[{"\$or":[{"_id":{"\$not":{"\$ne":"IMG841"}}}], [{"_id":{"\$ne":"IMG1924"}}]}]}	<i>_id, imagen y descriptor</i>	1				

Tabla 4.14 Comparación del esquema de fragmentación híbrida estática con fragmentación híbrida dinámica, segundo caso

Nombre	Predicado	Atributos	Nro. de documentos	Sitio origen	Sitio destino	Costo H-V	Costo V-H
<b>Colección de la base de datos HITO a fragmentar:</b>							
<i>multimedia_records</i>	N/A	<i>_id, nombre, descripción, imagen,</i>	1870	1	N/A	N/A	N/A

Nombre	Predicado	Atributos	Nro. de documentos	Sitio origen	Sitio destino	Costo H-V	Costo V-H
		<i>validado, descriptor y tipo</i>					
<b>Fragmentación híbrida estática</b>							
<i>multimedia_records_1</i>	{"\$and":[{"validado":{"\$eq":null}}]}	<i>_id, imagen y descriptor</i>	1843	1	2	205,341.259	205,341.259
<i>multimedia_records_2</i>	{"\$and":[{"validado":{"\$eq":null}}]}	<i>_id, nombre, descripción, validado y tipo</i>	1843				
<i>multimedia_records_3</i>	{"\$and":[{"\$or":[{"validado":{"\$not":{"\$eq":null}}}],{"validado":{"\$ne":null}}]}	<i>_id, imagen y descriptor</i>	27				
<i>multimedia_records_4</i>	{"\$and":[{"\$or":[{"validado":{"\$not":{"\$eq":null}}}],{"validado":{"\$ne":null}}]}	<i>_id, nombre, descripción, validado y tipo</i>	27				
<b>Fragmentación híbrida dinámica, segundo caso</b>							
<i>multimedia_records_1_1</i>	{"\$and":[{"_id":{"\$ne":"IMG1909"}}]}	<i>_id, nombre, descripción, validado y tipo</i>	1842	2	1	300,711.34	300,711.34
<i>multimedia_records_1_2</i>	{"\$and":[{"\$or":[{"_id":{"\$not":{"\$ne":"IMG1909"}}}],{"_id":{"\$ne":"IMG449"}}]}	<i>_id, nombre, descripción, validado y tipo</i>	1				
<i>multimedia_records_2_1</i>	{"\$and":[{"_id":{"\$ne":{"\$oid":"615e997af0f0e30bc46132ad"}}}]}	<i>_id, imagen y descriptor</i>	1842			304,272.85	304,272.85

Nombre	Predicado	Atributos	Nro. de documentos	Sitio origen	Sitio destino	Costo H-V	Costo V-H
<i>multimedia_records_2_2</i>	<code>{"\$and":[{"\$or":[{"_id":{"\$not":{"\$ne":{"\$oid":"615e997af0f0e30bc46132ad"}}}}}], {"_id":{"\$ne":"IMG768"}}}]}</code>	<i>_id, imagen y descriptor</i>	1				
<i>multimedia_records_3_1</i>	<code>{"\$and":[{"_id":{"\$ne":"IMG1930"}}]}</code>	<i>_id, nombre, descripción, validado y tipo</i>	26			78,047.62	78,047.62
<i>multimedia_records_3_2</i>	<code>{"\$and":[{"\$or":[{"_id":{"\$not":{"\$ne":"IMG1930"}}}], {"_id":{"\$ne":"IMG841"}}]}</code>	<i>_id, nombre, descripción, validado y tipo</i>	1				

Tabla 4.15 Comparación del esquema de fragmentación híbrida dinámica caso 1 y 2

Nombre del fragmento	Predicado	Atributos	Nro. de documentos	Costo H-V	Costo V-H
<b>Primer caso, fragmentación híbrida dinámica</b>					
<i>multimedia_records_1</i>	<code>{"\$and":[{"validado":{"\$eq":null}}]}</code>	<i>_id, nombre, descripción, validado y tipo</i>	1843	264,566.76	264,566.76
<i>multimedia_records_2_1</i>	<code>{"\$and":[{"_id":{"\$ne":"IMG1631"}}]}</code>	<i>_id, imagen y descriptor</i>	1842	157,170.17	157,170.17
<i>multimedia_records_2_2</i>	<code>{"\$and":[{"\$or":[{"_id":{"\$not":{"\$ne":"IMG1631"}}}], {"_id":{"\$ne":{"\$oid":"615e9812f0f0e30bc46131b5"}}}]}</code>	<i>_id, imagen y descriptor</i>	1		
<i>multimedia_records_3_1</i>	<code>{"\$and":[{"tipo":{"\$ne":"building"}}]}</code>	<i>_id, nombre, descripción,</i>	13		

Nombre del fragmento	Predicado	Atributos	Nro. de documentos	Costo H-V	Costo V-H
		<i>validado y tipo</i>		302,599.14	302,599.14
<i>multimedia_records_3_2</i>	{ "\$and": [ { \$or: [ { "tipo": { \$not: { "\$ne": "building" } } } ] }, { "tipo": { "\$ne": "event" } } ] }	<i>_id, nombre, descripción, validado y tipo</i>	14		
<i>multimedia_records_4_1</i>	{ "\$and": [ { "_id": { "\$ne": "IMG841" } } ] }	<i>_id, imagen y descriptor</i>	26		
<i>multimedia_records_4_2</i>	{ "\$and": [ { \$or: [ { "_id": { \$not: { "\$ne": "IMG841" } } } ] }, { "_id": { "\$ne": "IMG1924" } } ] }	<i>_id, imagen y descriptor</i>	1	302,599.14	302,599.14
<b>Segundo caso, fragmentación híbrida dinámica</b>					
<i>multimedia_records_1_1</i>	{ "\$and": [ { "_id": { "\$ne": "IMG1909" } } ] }	<i>_id, nombre, descripción, validado y tipo</i>	1842		
<i>multimedia_records_1_2</i>	{ "\$and": [ { \$or: [ { "_id": { \$not: { "\$ne": "IMG1909" } } } ] }, { "_id": { "\$ne": "IMG449" } } ] }	<i>_id, nombre, descripción, validado y tipo</i>	1	300,711.34	300,711.34
<i>multimedia_records_2_1</i>	{ "\$and": [ { "_id": { "\$ne": { "\$oid": "615e997af0f0e30bc46132ad" } } } ] }	<i>_id, imagen y descriptor</i>	1842		
<i>multimedia_records_2_2</i>	{ "\$and": [ { \$or: [ { "_id": { \$not: { "\$ne": { "\$oid": "615e997af0f0e30bc46132ad" } } } ] }, { "_id": { "\$ne": "IMG768" } } ] }	<i>_id, imagen y descriptor</i>	1	304,272.85	304,272.85
<i>multimedia_records_3_1</i>	{ "\$and": [ { "_id": { "\$ne": "IMG1930" } } ] }	<i>_id, nombre, descripción, validado y tipo</i>	26		
<i>multimedia_records_3_2</i>	{ "\$and": [ { \$or: [ { "_id": { \$not: { "\$ne": "IMG1930" } } } ] }, { "_id": { "\$ne": "IMG841" } } ] }	<i>_id, nombre, descripción, validado y tipo</i>	1	78,047.62	78,047.62

## 4.5 Comparación del rendimiento de la metodología propuesta

En esta sección, se presentan los resultados obtenidos al comparar el método de fragmentación híbrida dinámica propuesto con los trabajos de Castro-Medina et al. [9] y Rodríguez-Mazahua et al. [68], con el objetivo de medir la eficacia de este proyecto.

Para lograr la comparación, se utilizó la carga de trabajo de 100 operaciones presentada anteriormente en la Tabla 4.1 (Operaciones de la carga de trabajo para la fragmentación híbrida estática), donde se analizó cada una de las operaciones con los esquemas de fragmentación. Para esto, primero se aplicó la fragmentación horizontal, seguida de la vertical y, por último, la fragmentación híbrida estática.

Como se visualiza en la Figura 4.80, en la partición horizontal se formaron dos fragmentos los cuales son: *multimedia\_records\_1* y *multimedia\_records\_3*; “*multimedia\_records\_2*” no se tiene porque al estar vacío no se forma. Como se ha mencionado, la fragmentación horizontal solo considera los predicados en cada operación realizada, dicho lo anterior, para realizar la comparación en la horizontal se utiliza el predicado con traslape ( $\{\$and:\{\{"validado":\{"$eq":null\}\}\}$  y  $\{\$and:\{\{\$or:\{\{"validado":\{\$not:\{\{"$eq":null\}\}\}\}\}\}\}$ ,  $\{"validado":\{"$ne":null\}\}\}$ ) porque en ocasiones, hay más de un fragmento que comparte el mismo predicado y afecta en el resultado de la operación realizada, por consiguiente, el predicado con traslape permite resolver el problema considerando primero los predicados con mayor costo. Así mismo, ambos fragmentos formados en la horizontal conservan todos los atributos de la colección *multimedia\_records* (*\_id*, *nombre*, *descripción*, *imagen*, *validado*, *descriptor* y *tipo*), la diferencia entre cada fragmento es el número de documentos, el primero contiene 1843 y el segundo 27; además coincidió en que ambos fragmentos se asignan al sitio 2.

Cuando el fragmento se asigna al sitio donde se encuentra la base de datos, se agrega sobre la misma, pero si el sitio es diferente y no existe la base de datos, en el caso de MongoDB se crea la base de datos y los nuevos fragmentos.

Fragmento	Predicado	Predicado traslape	Sitio	Número de tuplas	Valor de desempeño
multimedia_records_1	{"validado":{"\$eq":null}}	{\$and:[{"validado":{"\$eq":null}}]}	192.168.1.75	1843	46075.0
multimedia_records_3	{"validado":{"\$ne":null}}	{\$and:[{"or":[{"validado":{"\$not":{"\$eq":null}}}],{"validado":{"\$ne":null}}]}	192.168.1.75	27	729.0

0%

Let's do it

Figura 4.80 Esquema de fragmentación horizontal estática

Por otra parte, el resultado del esquema de fragmentación vertical se presenta en la Figura 4.81. Esta fragmentación se basa en los atributos, por lo tanto, “*multimedia\_records\_1*” se conforma del *\_id*, *imagen* y *descriptor*, mientras que, “*multimedia\_records\_2*” tiene el resto de los atributos (*\_id*, *descripción*, *tipo*, *validado*, *nombre*). Aunque los atributos estén separados, al momento de realizar cualquier operación y se requiera de un atributo que no está en un fragmento, mediante reuniones se recupera la información completa, es por eso que, cada fragmento tiene todas las tuplas (1870) aunque los atributos se encuentren separados, es importante que cada fragmento contenga su identificador; este es único en cada documento a pesar de estar separados los atributos.

En la fragmentación híbrida se considera el esquema mostrado anteriormente en la Figura 4.25, donde se produjeron cuatro fragmentos cada uno con sus respectivos predicados con traslape y atributos.

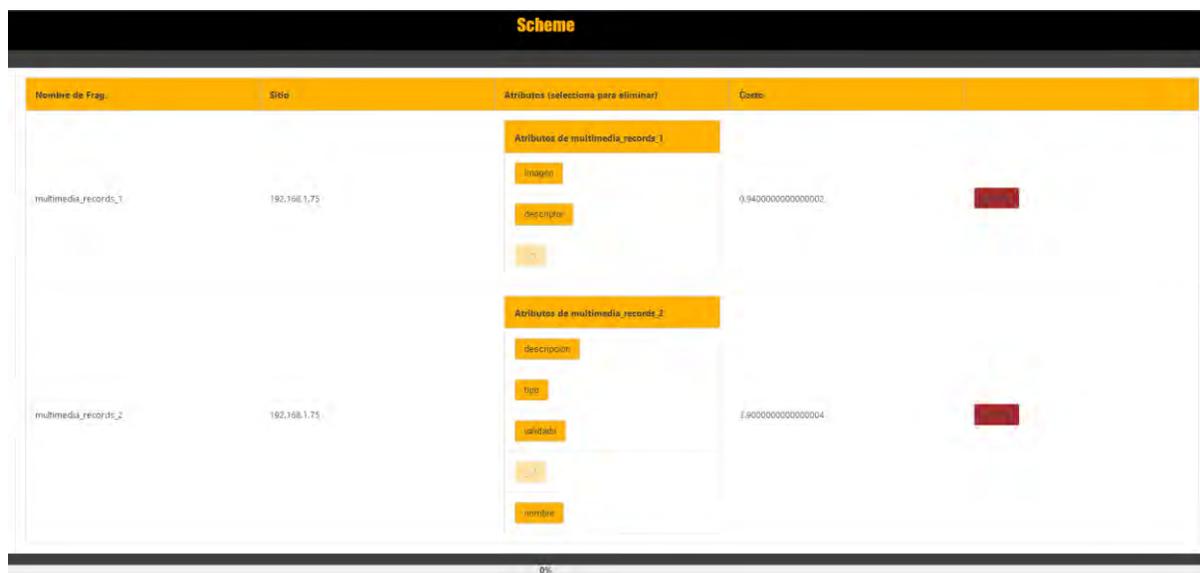


Figura 4.81 Esquema de fragmentación vertical estática

Una vez terminado de describir cada uno de los esquemas de fragmentación, se realizan las comparaciones, donde:

1. Se calcula el costo del esquema de fragmentación horizontal, vertical e híbrida con base en cada operación.
2. Se obtienen las tuplas y atributos irrelevantes por cada tipo de fragmentación, de acuerdo a la operación.

A continuación, en la Tabla 4.16 se presentan las 100 operaciones de la carga de trabajo contempladas en ambas comparaciones.

Tabla 4.16 Operaciones de la carga de trabajo para realizar la comparación

NRO.	OPERACIÓN
q1	db.multimedia_records.updateMany({"validado":{"\$eq:null"}}, {"\$set":{"validado":null}})
q2	db.multimedia_records.find({"tipo":{"\$eq:"personal"}}, {"validado":1})
q3	db.multimedia_records.deleteMany({"descripcion":{"\$eq:"."}})
q4	db.multimedia_records.find({"validado":{"\$ne:null"}}, {"validado":1,"tipo":1})
q5	db.multimedia_records.insertOne({"nombre":"LabIng06"})
q6	db.multimedia_records.insertOne({"id":"IMG1696"})
q7	db.multimedia_records.updateMany({"nombre":{"\$eq:"ALICIA2"}}, {"\$set":{"nombre":"VELPERLE"}})
q8	db.multimedia_records.updateMany({"descripcion":{"\$eq:"."}}, {"\$set":{"descripcion":"..."}})
q9	db.multimedia_records.deleteMany({"validado":{"\$eq:null"}})

NRO.	OPERACIÓN
q10	db.multimedia_records.insertOne({" id":"IMG385"})
q11	db.multimedia_records.find({"nombre":{"\$ne:"13MAYO95"}}, {"descripcion":1})
q12	db.multimedia_records.find({"descripcion":{"\$eq:"."}}, {"tipo":1})
q13	db.multimedia_records.find({"descripcion":{"\$eq:"Caricatura a mano"}}, {"nombre":":1, "validado":1, " id":1,"tipo":1,"descripcion":1})
q14	db.multimedia_records.updateMany({" id":{"\$eq:"IMG1842"}}, {"\$set":{" id":"IMG402"}})
q15	db.multimedia_records.updateMany({" id":{"\$ne:"IMG1792"}}, {"\$set":{" id":"IMG385"}})
q16	db.multimedia_records.find({"validado":{"\$ne:null}}, {"tipo":1,"nombre":1})
q17	db.multimedia_records.find({"tipo":{"\$eq:"personal"}}, {" id":1,"tipo":1,"validado":1,"nombre":1})
q18	db.multimedia_records.updateMany({"descripcion":{"\$eq:"..."}}, {"\$set":{"descripcion":"."}})
q19	db.multimedia_records.find({"validado":{"\$ne:null}}, {"tipo":1, " id":1,"descripcion":1,"nombre":1, "validado":1})
q20	db.multimedia_records.find({"_id":{"\$eq":{"\$oid":"615e96ddf0f0e30bc46130cf"}}, {"_id":1,"tipo":1, "validado":1,"nombre":1})
q21	db.multimedia_records.find({"nombre":{"\$eq:"CREA5-3"}}, {"tipo":1,"descripcion":1,"validado":1})
q22	db.multimedia_records.insertOne({"nombre":"Desfile1"})
q23	db.multimedia_records.deleteMany({"tipo":{"\$eq:"personal"}})
q24	db.multimedia_records.insertOne({"descripcion":"Ramón Matías López."})
q25	db.multimedia_records.insertOne({"nombre":"zepperri"})
q26	db.multimedia_records.insertOne({" id": ObjectId("615e97a3f0f0e30bc4613125"))
q27	db.multimedia_records.updateMany({"validado":{"\$eq:null}}, {"\$set":{"validado":null}})
q28	db.multimedia_records.updateMany({"descripcion":{"\$eq:"Alvarez Guzman Rafael"}}, {"\$set":{"descripcion":"."}})
q29	db.multimedia_records.updateMany({"descripcion":{"\$eq:"Caricatura a mano"}}, {"\$set":{"descripcion":"."}})
q30	db.multimedia_records.find({"validado":{"\$eq:null}}, {"nombre":1})
q31	db.multimedia_records.find({"_id":{"\$ne":{"\$oid":"615e97d3f0f0e30bc461315b"}}, {"tipo":1, "descripcion":1,"validado":1, " id":1})
q32	db.multimedia_records.find({"tipo":{"\$ne:"building"}}, {" id":1})
q33	db.multimedia_records.find({"tipo":{"\$eq:"event"}}, {" id":1,"nombre":1,"descripcion":1})
q34	db.multimedia_records.deleteMany({"validado":{"\$ne:null}})
q35	db.multimedia_records.insertOne({"nombre":"CONVE94"})
q36	db.multimedia_records.find({"validado":{"\$ne:null}}, {"_id":1,"validado":1,"descripcion":1, "nombre":1,"tipo":1})
q37	db.multimedia_records.updateMany({"tipo":{"\$eq:"event"}}, {"\$set":{"tipo":"event"}})
q38	db.multimedia_records.updateMany({" id":{"\$eq:"IMG144"}}, {"\$set":{" id":"IMG1769"}})
q39	db.multimedia_records.find({"tipo":{"\$ne:"event"}}, {"descripcion":1,"tipo":1, " id":1,"nombre":1, "validado":1})
q40	db.multimedia_records.deleteMany({" id":{"\$eq:"IMG648"}})
q41	db.multimedia_records.insert({"validado":null})
q42	db.multimedia_records.insertOne({" id": ObjectId("615e988bf0f0e30bc46131da"))
q43	db.multimedia_records.deleteMany({"descripcion":{"\$eq:"."}})
q44	db.multimedia_records.insertOne({" id": ObjectId("615e951a92bd5f6ac041f471"))
q45	db.multimedia_records.deleteMany({"descripcion":{"\$ne:"..."}})
q46	db.multimedia_records.find({"descripcion":{"\$ne:"."}}, {"validado":1,"descripcion":1,"tipo":1,

NRO.	OPERACIÓN
	" id":1})
q47	db.multimedia_records.insertOne({"tipo":"personal"})
q48	db.multimedia_records.updateMany({"validado":{"\$ne:null"}},{"\$set":{"validado":null}})
q49	db.multimedia_records.updateMany({"_id":{"\$eq":{"\$oid":"615e9965f0f0e30bc461327d"}}, {"\$set":{" id":{"\$oid":"615e9944f0f0e30bc461325e"}}})
q50	db.multimedia_records.deleteMany({"nombre":{"\$eq:"DLAzteca"}})
q51	db.multimedia_records.updateMany({" id":{"\$eq:"IMG1825"}}, {"\$set":{" id":"IMG364"}})
q52	db.multimedia_records.deleteMany({" id":{"\$eq:"IMG1817"}})
q53	db.multimedia_records.updateMany({"nombre":{"\$eq:"CREA58"}}, {"\$set":{"nombre": "INAUGP~5"}})
q54	db.multimedia_records.deleteMany({"validado":{"\$ne:null}})
q55	db.multimedia_records.deleteMany({"nombre":{"\$ne:"BAZMENMM"}})
q56	db.multimedia_records.updateMany({"tipo":{"\$eq:"building"}}, {"\$set":{"tipo":"building"}})
q57	db.multimedia_records.insertOne({"descripcion":"."})
q58	db.multimedia_records.find({"tipo":{"\$eq:"event"}}, {"nombre":1})
q59	db.multimedia_records.deleteMany({"descripcion":{"\$ne:"."}})
q60	db.multimedia_records.updateMany({"nombre":{"\$eq:"Jimsanci"}}, {"\$set":{"nombre":"CREA5-8"}})
q61	db.multimedia_records.find({"tipo":{"\$ne:"personal"}}, {"nombre":1,"descripcion":1,"validado":1, "tipo":1," id":1})
q62	db.multimedia_records.insertOne({" id": ObjectId ("615e98fdf0f0e30bc461320c")})
q63	db.multimedia_records.deleteMany({"tipo":{"\$eq:"event"}})
q64	db.multimedia_records.find({"validado":{"\$eq:null"}}, {" id":1,"descripcion":1})
q65	db.multimedia_records.deleteMany({" id":{"\$ne:"IMG1586"}})
q66	db.multimedia_records.find({"descripcion":{"\$ne:"."}}, {"nombre":1, " id":1,"descripcion":1})
q67	db.multimedia_records.updateMany({"validado":{"\$ne:null"}}, {"\$set":{"validado":null}})
q68	db.multimedia_records.updateMany({"descripcion":{"\$ne:"Aguilar Aquino Alva María del Carmen"}}, {"\$set":{"descripcion":"Caricatura a mano"}})
q69	db.multimedia_records.insertOne({"validado":null})
q70	db.multimedia_records.updateMany({"tipo":{"\$eq:"event"}}, {"\$set":{"tipo":"event"}})
q71	db.multimedia_records.updateMany({"descripcion":{"\$ne:"."}}, {"\$set":{"descripcion":"."}})
q72	db.multimedia_records.insertOne({"nombre":"Perio03"})
q73	db.multimedia_records.updateMany({"tipo":{"\$eq:"personal"}}, {"\$set":{"tipo":"personal"}})
q74	db.multimedia_records.updateMany({"tipo":{"\$eq:"equipment"}}, {"\$set":{"tipo":"event"}})
q75	db.multimedia_records.updateMany({" id":{"\$ne:"IMG1820"}}, {"\$set":{" id":"IMG1558"}})
q76	db.multimedia_records.updateMany({"validado":{"\$ne:null"}}, {"\$set":{"validado":null}})
q77	db.multimedia_records.deleteMany({"tipo":{"\$eq:"event"}})
q78	db.multimedia_records.find({"nombre":{"\$eq:"9DIC91"}}, {"nombre":1})
q79	db.multimedia_records.insertOne({"nombre":"MARROJLB"})
q80	db.multimedia_records.insertOne({"descripcion":"."})
q81	db.multimedia_records.deleteMany({"validado":{"\$eq:null}})
q82	db.multimedia_records.deleteMany({" id":{"\$ne:"IMG1574"}})
q83	db.multimedia_records.find({"validado":{"\$ne:null"}}, {" id":1,"nombre":1})
q84	db.multimedia_records.updateMany({" id":{"\$ne:"IMG1797"}}, {"\$set":{" id":"IMG1820"}})
q85	db.multimedia_records.insertOne({" id": ObjectId ("615e94fe92bd5f6ac041f451")})

NRO.	OPERACIÓN
q86	db.multimedia_records.deleteMany({"validado":{"\$eq:null}})
q87	db.multimedia_records.insertOne({"descripcion":"Alvarado Paz José Luis"})
q88	db.multimedia_records.deleteMany({"descripcion":{"\$eq:"."}})
q89	db.multimedia_records.updateMany({"validado":{"\$eq:null"}}, {"\$set":{"validado":null}})
q90	db.multimedia_records.updateMany({"tipo":{"\$ne:"event"}}, {"\$set":{"tipo":"event"}})
q91	db.multimedia_records.find({"descripcion":{"\$ne:"Caricatura a mano "}}, {"tipo":1, "descripcion":1, "id":1, "validado":1, "nombre":1})
q92	db.multimedia_records.deleteMany({"id":{"\$ne:"IMG1815"}})
q93	db.multimedia_records.insertOne({"nombre":"LabMec05"})
q94	db.multimedia_records.insertOne({"validado":null})
q95	db.multimedia_records.deleteMany({"descripcion":{"\$ne:"."}})
q96	db.multimedia_records.insertOne({"nombre":"SN198xxxx-103(JMH)"})
q97	db.multimedia_records.updateMany({"_id":{"\$ne":{"\$oid":"615e9680f0f0e30bc4613071"}}}, {"\$set":{"id":{"\$oid":"615e97cdf0f0e30bc461314c"}}})
q98	db.multimedia_records.find({"descripcion":{"\$eq:"Juegos Prenacionales de Teconológicos, en las disciplinas de atletismo, natación, tenis, ajedrez, de las zonas IX,X,XI, estando presentes los tecnológicos de: Apizaco, Cancún, Campeche, Cuautla, Chetumal, Itsmo, Mérida, Minatitlan, Oaxaca, Puebla, Tapachula, Tehuacan, Tuxtla Gutierrez, Veracruz, Villahermosa y el anfitrión Orizaba."}}, {"tipo":1, "id":1, "descripcion":1})
q99	db.multimedia_records.deleteMany({"tipo":{"\$ne:"event"}})
q100	db.multimedia_records.updateMany({"descripcion":{"\$eq:"."}}, {"\$set":{"descripcion":"..."}})

Se realiza la comparación con base en el siguiente criterio:

Aspecto	Descripción
<b>qn</b>	Representa el número de operación en la carga de trabajo
<b>OPERACIÓN</b>	Cualquier operación ( <i>insert, find, update, delete</i> )
<b>SITIO</b>	Sitio de donde proviene la operación, más no el sitio del fragmento que satisface dicha operación
<b>EFV</b>	Esquema de Fragmentación Vertical
<b>EFH</b>	Esquema de Fragmentación Horizontal
<b>EFHI</b>	Esquema de Fragmentación Híbrido
<b>CE</b>	Costo de ejecución en bytes
<b>TI</b>	Tuplas irrelevantes
<b>AI</b>	Atributos irrelevantes

Se presentan los resultados obtenidos de la primera comparación realizada con el trabajo de Castro-Medina et al. [9] (Tabla 4.17).

Tabla 4.17 Resultado de la comparación con el trabajo de Castro-Medina et al. [9].

NRO.	SITIO	EFV			EFH			EFHI		
		CE	TI	AI	CE	TI	AI	CE	TI	AI
q1	1	60,000	27	40,000	11,058	0	120,000	35,529	0	40,000
q2	1	40,000	996	30,000	1,748	996	110,000	20,874	996	30,000
q3	1	80,000	565	40,000	5,220	565	120,000	42,610	565	40,000
q4	1	40,000	1,843	30,000	54	0	110,000	20,027	1,843	30,000
q5	1	40,000	0	0	4	0	0	20,002	0	0
q6	1	40,000	0	0	4	0	0	20,002	0	0
q7	1	60,000	1,869	40,000	6	1,869	120,000	30,003	1,869	40,000
q8	1	60,000	565	40,000	7,830	565	120,000	33,915	565	40,000
q9	1	80,000	27	40,000	7,372	0	120,000	43,686	0	40,000
q10	1	40,000	0	0	4	0	0	20,002	0	0
q11	1	40,000	1	30,000	3,738	1	110,000	21,869	1	30,000
q12	1	40,000	565	30,000	2,610	565	110,000	21,305	565	30,000
q13	1	100,000	1,665	0	410	1,638	80,000	50,205	1,665	0
q14	1	120,000	1,869	120,000	6	1,842	120,000	60,003	1,869	120,000
q15	1	120,000	1	120,000	11,214	1	120,000	65,607	1	120,000
q16	1	60,000	1,843	20,000	54	0	110,000	30,027	1,843	20,000
q17	1	80,000	996	10,000	1,748	996	90,000	40,874	996	10,000
q18	1	60,000	1,739	40,000	786	1,739	120,000	30,393	1,739	40,000
q19	1	100,000	1,843	0	54	0	80,000	50,027	1,843	0
q20	1	160,000	1,869	10,000	2	1,842	90,000	80,001	1,869	10,000
q21	1	80,000	1,869	10,000	2	1,842	90,000	40,001	1,869	10,000
q22	1	40,000	0	0	4	0	0	20,002	0	0
q23	1	80,000	996	40,000	3,496	996	120,000	41,748	996	40,000
q24	1	40,000	0	0	4	0	0	20,002	0	0
q25	1	40,000	0	0	4	0	0	20,002	0	0
q26	1	40,000	0	0	4	0	0	20,002	0	0
q27	1	60,000	27	40,000	11,058	0	120,000	35,529	0	40,000
q28	1	60,000	1,869	40,000	6	1,842	120,000	30,003	1,869	40,000
q29	1	60,000	1,665	40,000	1,230	1,638	120,000	30,615	1,665	40,000
q30	1	40,000	27	30,000	3,686	0	110,000	21,843	27	30,000
q31	1	80,000	0	10,000	3,740	0	90,000	41,870	0	10,000
q32	1	40,000	168	30,000	3,404	168	110,000	21,702	168	30,000
q33	1	80,000	1,146	10,000	1,448	1,146	90,000	40,724	1,146	10,000
q34	1	80,000	1,843	40,000	108	0	120,000	40,054	0	40,000
q35	1	40,000	0	0	4	0	0	20,002	0	0
q36	1	100,000	1,843	0	54	0	80,000	50,027	1,843	0
q37	1	60,000	1,146	40,000	4,344	1,146	120,000	32,172	1,146	40,000
q38	1	120,000	1,869	120,000	6	1,842	120,000	60,003	1,869	120,000
q39	1	100,000	724	0	2,292	724	80,000	51,146	724	0
q40	1	80,000	1,869	120,000	4	1,842	120,000	40,002	1,869	120,000

NRO.	SITIO	EFV			EFH			EFHI		
		CE	TI	AI	CE	TI	AI	CE	TI	AI
q41	1	40,000	0	0	4	0	0	20,002	0	0
q42	1	40,000	0	0	4	0	0	20,002	0	0
q43	1	80,000	565	40,000	5,220	565	120,000	42,610	565	40,000
q44	1	40,000	0	0	4	0	0	20,002	0	0
q45	1	80,000	131	40,000	6,956	131	120,000	43,478	131	40,000
q46	1	80,000	1,305	10,000	1,130	1,305	90,000	40,565	1,305	10,000
q47	1	40,000	0	0	4	0	0	20,002	0	0
q48	1	60,000	1,843	40,000	162	0	120,000	30,081	1,843	40,000
q49	1	120,000	1,869	120,000	6	1,842	120,000	60,003	1,869	120,000
q50	1	80,000	1,869	40,000	4	1,842	120,000	40,002	1,869	40,000
q51	2	60,000	1,869	120,000	3	1,842	120,000	30,002	1,869	120,000
q52	2	40,000	1,869	120,000	2	1,842	120,000	20,001	1,869	120,000
q53	2	30,000	1,869	40,000	3	1,842	120,000	15,002	1,869	40,000
q54	2	40,000	1,843	40,000	54	0	120,000	20,027	1,843	40,000
q55	2	40,000	1	40,000	3,738	1	120,000	21,869	1	40,000
q56	2	30,000	1,702	40,000	504	1,702	120,000	15,252	1,702	40,000
q57	2	20,000	0	0	2	0	0	10,001	0	0
q58	2	20,000	1,146	30,000	724	1,146	110,000	10,362	1,146	30,000
q59	2	40,000	1,305	40,000	1,130	1,305	120,000	20,565	1,305	40,000
q60	2	30,000	1,869	40,000	3	1,842	120,000	15,002	1,869	40,000
q61	2	50,000	874	0	996	874	80,000	25,498	874	0
q62	2	20,000	0	0	2	0	0	10,001	0	0
q63	2	40,000	1,146	40,000	1,448	1,146	120,000	20,724	1,146	40,000
q64	2	30,000	27	20,000	1,843	0	100,000	15,922	27	20,000
q65	2	40,000	1	120,000	3,738	1	120,000	21,869	1	120,000
q66	2	30,000	1,305	20,000	565	1,305	100,000	15,283	1,305	20,000
q67	2	30,000	1,843	40,000	81	0	120,000	15,041	0	40,000
q68	2	30,000	1	40,000	5,607	1	120,000	17,804	1	40,000
q69	2	20,000	0	0	2	0	0	10,001	0	0
q70	2	30,000	1,146	40,000	2,172	1,146	120,000	16,086	1,146	40,000
q71	2	30,000	1,305	40,000	1,695	1,305	120,000	15,848	1,305	40,000
q72	2	20,000	0	0	2	0	0	10,001	0	0
q73	2	30,000	996	40,000	2,622	996	120,000	16,311	996	40,000
q74	2	30,000	1,766	40,000	312	1,766	120,000	15,156	1,766	40,000
q75	2	60,000	1	120,000	5,607	1	120,000	32,804	1	120,000
q76	2	30,000	1,843	40,000	81	0	120,000	15,041	0	40,000
q77	2	40,000	1,146	40,000	1,448	1,146	120,000	20,724	1,146	40,000
q78	2	10,000	1,869	40,000	1	1,842	120,000	5,001	1,869	40,000
q79	2	20,000	0	0	2	0	0	10,001	0	0
q80	2	20,000	0	0	2	0	0	10,001	0	0
q81	2	40,000	27	40,000	3,686	0	120,000	21,843	0	40,000
q82	2	40,000	1	120,000	3,738	1	120,000	21,869	1	120,000
q83	2	30,000	1,843	20,000	27	0	100,000	15,014	0	20,000
q84	2	60,000	1	120,000	5,607	1	120,000	32,804	1	120,000
q85	2	20,000	0	0	2	0	0	10,001	0	0

NRO.	SITIO	EFV			EFH			EFHI		
		CE	TI	AI	CE	TI	AI	CE	TI	AI
q86	2	40,000	<b>27</b>	40,000	<b>3,686</b>	0	120,000	21,843	<b>0</b>	40,000
q87	2	10,000	<b>0</b>	<b>0</b>	<b>1</b>	<b>0</b>	<b>0</b>	5,001	<b>0</b>	<b>0</b>
q88	2	40,000	<b>565</b>	40,000	<b>2,610</b>	<b>565</b>	120,000	21,305	<b>565</b>	40,000
q89	2	30,000	<b>27</b>	120,000	<b>5,529</b>	<b>0</b>	120,000	17,765	<b>0</b>	120,000
q90	2	30,000	<b>724</b>	40,000	<b>3,438</b>	<b>724</b>	120,000	16,719	<b>724</b>	40,000
q91	2	50,000	<b>205</b>	<b>0</b>	<b>1,665</b>	<b>205</b>	80,000	25,833	<b>205</b>	<b>0</b>
q92	2	40,000	<b>1</b>	120,000	<b>3,738</b>	<b>1</b>	120,000	21,869	<b>1</b>	120,000
q93	2	20,000	<b>0</b>	<b>0</b>	<b>2</b>	<b>0</b>	<b>0</b>	10,001	<b>0</b>	<b>0</b>
q94	2	20,000	<b>0</b>	<b>0</b>	<b>2</b>	<b>0</b>	<b>0</b>	10,001	<b>0</b>	<b>0</b>
q95	2	40,000	<b>1,305</b>	40,000	<b>1,130</b>	<b>1,305</b>	120,000	20,565	<b>1,305</b>	40,000
q96	2	20,000	<b>0</b>	<b>0</b>	<b>2</b>	<b>0</b>	<b>0</b>	10,001	<b>0</b>	<b>0</b>
q97	2	60,000	<b>0</b>	40,000	<b>5,610</b>	<b>0</b>	120,000	32,805	<b>0</b>	120,000
q98	2	60,000	<b>1,869</b>	20,000	<b>2</b>	<b>1,842</b>	100,000	30,001	<b>1,869</b>	20,000
q99	2	40,000	<b>724</b>	40,000	<b>2,292</b>	<b>724</b>	120,000	21,146	<b>724</b>	40,000
q100	2	30,000	<b>565</b>	40,000	<b>3,915</b>	<b>565</b>	120,000	16,958	<b>565</b>	40,000

Con base en los resultados obtenidos en la primera comparación, el esquema horizontal es el que tiene un menor costo de ejecución en bytes, ya que no se considera el tamaño de los atributos, sin embargo, es el que tiene una mayor cantidad de acceso a atributos irrelevantes, por otro lado, para el caso de la fragmentación vertical se evita el acceso a atributos irrelevantes porque solo considera a los atributos que requiere en la operación realizada, y por lo tanto, accede a una mayor cantidad de tuplas irrelevantes. En el caso de la híbrida, al hacer la mezcla de la horizontal y vertical, contempla los predicados y atributos, teniendo un mejor resultado en las tuplas y atributos irrelevantes.

Por otra parte, si se realiza una comparación entre el esquema vertical con el híbrido, la fragmentación híbrida tiene menor costo de ejecución en bytes, tal como se presenta en la Tabla 4.18.

Tabla 4.18 Comparación de costo de ejecución esquema vertical con híbrido

NRO.	OPERACIÓN	EFV			EFHI		
		CE	TI	AI	CE	TI	AI
q5	db.multimedia_records.insertOne({"nombre":"LabIng06"})	40,000	0	0	<b>20,002</b>	0	0
q9	db.multimedia_records.deleteMany({"validado":{"\$eq:null"}})	80,000	27	40,000	<b>43,686</b>	0	40,000
q28	db.multimedia_records.updateMany({"descripcion":{"\$eq: "Alvarez Guzman Rafael"}}, {"\$set":{"descripcion":"."}})	60,000	1,869	40,000	<b>30,003</b>	1,869	40,000
q39	db.multimedia_records.find({"tipo":{"\$ne:"event"}}, {"descripcion":1,"tipo":1,"_id":1,"nombre":1,"validado":1})	100,000	724	0	<b>51,146</b>	724	0

Para fragmentar *multimedia\_records* con el trabajo de Rodríguez-Mazahua et al. [68] se utilizaron las siguientes consultas: q2, q4, q11, q12, q13, q16, q17, q19, q20, q21, q30, q31, q32, q33, q36, q39, q46, q58, q61, q64, q66, q78, q83, q91 y q98. El resultado obtenido de acuerdo a las consultas mencionadas anteriormente, es un esquema de fragmentación híbrida estática con 8 fragmentos (Tabla 4.19), de los cuales, los primeros cuatro fragmentos tienen los mismos atributos (*descripción*, *tipo*, *validado*, *\_id* y *nombre*) cada uno y los predicados son diferentes, mientras que el resto, tiene los atributos faltantes (*\_id*, *imagen* y *descriptor*); todo el esquema se aplicó al sitio 1.

Tabla 4.19 Esquema de fragmentación híbrida dinámica, con base en el trabajo de Rodríguez-Mazahua et al. [68]

Fragmento	Atributos	Predicado
<i>multimedia_records_1</i>	<i>descripción</i> , <i>tipo</i> , <i>validado</i> , <i>id</i> y <i>nombre</i>	{"tipo":{"\$eq:"personal"}}
<i>multimedia_records_2</i>	<i>descripción</i> , <i>tipo</i> , <i>validado</i> , <i>id</i> y <i>nombre</i>	{"tipo":{"\$eq:"equipment"}}
<i>multimedia_records_3</i>	<i>descripción</i> , <i>tipo</i> , <i>validado</i> , <i>id</i> y <i>nombre</i>	{"tipo":{"\$eq:"event"}}
<i>multimedia_records_4</i>	<i>descripción</i> , <i>tipo</i> , <i>validado</i> , <i>id</i> y <i>nombre</i>	{"tipo":{"\$eq:"building"}}
<i>multimedia_records_5</i>	<i>id</i> , <i>imagen</i> , <i>descriptor</i>	{"tipo":{"\$eq:"personal"}}
<i>multimedia_records_6</i>	<i>id</i> , <i>imagen</i> , <i>descriptor</i>	{"tipo":{"\$eq:"equipment"}}
<i>multimedia_records_7</i>	<i>id</i> , <i>imagen</i> , <i>descriptor</i>	{"tipo":{"\$eq:"event"}}
<i>multimedia_records_8</i>	<i>id</i> , <i>imagen</i> , <i>descriptor</i>	{"tipo":{"\$eq:"building"}}

A continuación, en la Tabla 4.20 se muestran los resultados obtenidos en la comparación del esquema de fragmentación híbrida propuesto en este trabajo con el de Rodríguez-Mazahua et al. [68].

Tabla 4.20 Resultado de la comparación con el trabajo de Rodríguez-Mazahua et al. [68]

<b>NRO.</b>	<b>Costo EFHI</b>	<b>CR (Esquema vertical)</b>	<b>Costo EFHI Rodríguez-Mazahua et al. [68]</b>	<b>CR (Esquema vertical)</b>
q1	<b>35,529</b>	<b>1</b>	62,765	4
q2	20,874	<b>1</b>	<b>10,437</b>	<b>1</b>
q3	42,610	<b>2</b>	<b>41,305</b>	4
q4	<b>20,027</b>	<b>1</b>	40,014	4
q5	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q6	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q7	<b>30,003</b>	<b>1</b>	60,002	4
q8	<b>33,915</b>	<b>1</b>	61,958	4
q9	43,686	<b>2</b>	<b>41,843</b>	4
q10	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q11	<b>21,869</b>	<b>1</b>	40,935	4
q12	<b>21,305</b>	<b>1</b>	40,653	4
q13	<b>50,205</b>	<b>1</b>	100,103	4
q14	<b>60,003</b>	<b>2</b>	120,002	8
q15	<b>65,607</b>	<b>2</b>	122,804	8
q16	<b>30,027</b>	<b>1</b>	60,014	4
q17	40,874	<b>1</b>	<b>20,437</b>	<b>1</b>
q18	<b>30,393</b>	<b>1</b>	60,197	4
q19	<b>50,027</b>	<b>1</b>	100,014	4
q20	<b>80,001</b>	<b>2</b>	<b>80,001</b>	4
q21	<b>40,001</b>	<b>1</b>	80,001	4
q22	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q23	41,748	2	<b>10,874</b>	<b>1</b>
q24	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q25	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q26	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q27	<b>35,529</b>	<b>1</b>	62,765	4
q28	<b>30,003</b>	<b>1</b>	60,002	4
q29	<b>30,615</b>	<b>1</b>	60,308	4
q30	<b>21,843</b>	<b>1</b>	40,922	4
q31	<b>41,870</b>	<b>1</b>	80,935	4
q32	<b>21,702</b>	<b>1</b>	40,851	4

<b>NRO.</b>	<b>Costo EFHI</b>	<b>CR (Esquema vertical)</b>	<b>Costo EFHI Rodríguez-Mazahua et al. [68]</b>	<b>CR (Esquema vertical)</b>
q33	40,724	<b>1</b>	<b>20,362</b>	<b>1</b>
q34	40,054	<b>2</b>	<b>40,027</b>	4
q35	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q36	<b>50,027</b>	<b>1</b>	100,014	4
q37	32,172	<b>1</b>	<b>16,086</b>	<b>1</b>
q38	<b>60,003</b>	<b>2</b>	120,002	8
q39	<b>51,146</b>	<b>1</b>	100,573	4
q40	<b>40,002</b>	<b>2</b>	80,001	8
q41	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q42	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q43	42,610	<b>2</b>	<b>41,305</b>	4
q44	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q45	43,478	<b>2</b>	<b>41,739</b>	4
q46	<b>40,565</b>	<b>1</b>	80,283	4
q47	20,002	<b>1</b>	<b>10,001</b>	<b>1</b>
q48	<b>30,081</b>	<b>1</b>	60,041	4
q49	<b>60,003</b>	<b>2</b>	120,002	8
q50	40,002	<b>2</b>	<b>40,001</b>	4
q51	<b>30,002</b>	<b>2</b>	240,003	8
q52	<b>20,001</b>	<b>2</b>	160,002	8
q53	<b>15,002</b>	<b>1</b>	120,003	4
q54	<b>20,027</b>	<b>2</b>	80,054	4
q55	<b>21,869</b>	<b>2</b>	83,738	4
q56	<b>15,252</b>	<b>1</b>	30,504	<b>1</b>
q57	<b>10,001</b>	<b>1</b>	20,002	<b>1</b>
q58	<b>10,362</b>	<b>1</b>	20,724	<b>1</b>
q59	<b>20,565</b>	<b>2</b>	81,130	4
q60	<b>15,002</b>	<b>1</b>	120,003	4
q61	<b>25,498</b>	<b>1</b>	200,996	4
q62	<b>10,001</b>	<b>1</b>	20,002	<b>1</b>
q63	<b>20,724</b>	<b>2</b>	21,448	<b>1</b>
q64	<b>15,922</b>	<b>1</b>	121,843	4
q65	<b>21,869</b>	<b>2</b>	163,738	8
q66	<b>15,283</b>	<b>1</b>	120,565	4
q67	<b>15,041</b>	<b>1</b>	120,081	4
q68	<b>17,804</b>	<b>1</b>	125,607	4
q69	<b>10,001</b>	<b>1</b>	20,002	<b>1</b>
q70	<b>16,086</b>	<b>1</b>	32,172	<b>1</b>
q71	<b>15,848</b>	<b>1</b>	121,695	4
q72	<b>10,001</b>	<b>1</b>	20,002	<b>1</b>

<b>NRO.</b>	<b>Costo EFHI</b>	<b>CR (Esquema vertical)</b>	<b>Costo EFHI Rodríguez-Mazahua et al. [68]</b>	<b>CR (Esquema vertical)</b>
q73	<b>16,311</b>	<b>1</b>	32,622	<b>1</b>
q74	<b>15,156</b>	<b>1</b>	30,312	<b>1</b>
q75	<b>32,804</b>	<b>2</b>	245,607	8
q76	<b>15,041</b>	<b>1</b>	120,081	4
q77	<b>20,724</b>	<b>2</b>	21,448	<b>1</b>
q78	<b>5,001</b>	<b>1</b>	40,001	4
q79	<b>10,001</b>	<b>1</b>	20,002	<b>1</b>
q80	<b>10,001</b>	<b>1</b>	20,002	<b>1</b>
q81	<b>21,843</b>	<b>2</b>	83,686	4
q82	<b>21,869</b>	<b>2</b>	163,738	8
q83	<b>15,014</b>	<b>1</b>	120,027	4
q84	<b>32,804</b>	<b>2</b>	245,607	8
q85	<b>10,001</b>	<b>1</b>	20,002	<b>1</b>
q86	<b>21,843</b>	<b>2</b>	83,686	4
q87	<b>5,001</b>	<b>1</b>	10,001	<b>1</b>
q88	<b>21,305</b>	<b>2</b>	82,610	4
q89	<b>17,765</b>	<b>1</b>	125,529	4
q90	<b>16,719</b>	<b>1</b>	123,438	4
q91	<b>25,833</b>	<b>1</b>	201,665	4
q92	<b>21,869</b>	<b>2</b>	163,738	8
q93	<b>10,001</b>	<b>1</b>	20,002	<b>1</b>
q94	<b>10,001</b>	<b>1</b>	20,002	<b>1</b>
q95	<b>20,565</b>	<b>2</b>	81,130	4
q96	<b>10,001</b>	<b>1</b>	20,002	<b>1</b>
q97	<b>32,805</b>	<b>2</b>	245,610	8
q98	<b>30,001</b>	<b>1</b>	240,002	4
q99	<b>21,146</b>	<b>2</b>	82,292	4
q100	<b>16,958</b>	<b>1</b>	123,915	4

Como se observa en la Tabla 4.20, existe una diferencia entre los costos híbridos obtenidos en cada esquema y operación realizada. El esquema híbrido obtenido con el método propuesto tiene un menor costo que la técnica presentada en Rodríguez-Mazahua et al. [68] en el 77% de las operaciones. El esquema de fragmentación híbrida, presentado en la Tabla 4.19, produjo ocho fragmentos híbridos y la mayoría de las operaciones requieren de al menos cuatro reuniones, provocando que aumente el costo, mientras que el método propuesto en esta investigación utiliza menos reuniones en cada operación, reduciendo así el costo.

En la Figura 4.82 se muestra de manera gráfica el resultado obtenido de la comparación realizada anteriormente. Como se observa, el número de fragmentos formados en la fragmentación híbrida (ya sea estática o dinámica) dependerá de las operaciones que se realicen, además, cada fragmento, independientemente de los atributos que incluya, tendrán diferentes predicados y documentos.

Por tal motivo, la selección del tipo de fragmentación a aplicar dependerá de lo que requiera optimizar el administrador de la base de datos, es decir, si el administrador busca que las operaciones accedan a menos documentos irrelevantes, la fragmentación horizontal es la más apropiada, pero si el DBA está interesado en reducir el acceso a atributos irrelevantes, la vertical es la mejor opción. Cuando las operaciones involucran subconjuntos de documentos y atributos, la fragmentación híbrida es la mejor alternativa. El método propuesto en este trabajo de investigación es una mejora para la fragmentación híbrida estática, porque permite realizar constantemente nuevas refragmentaciones sin la intervención del DBA.

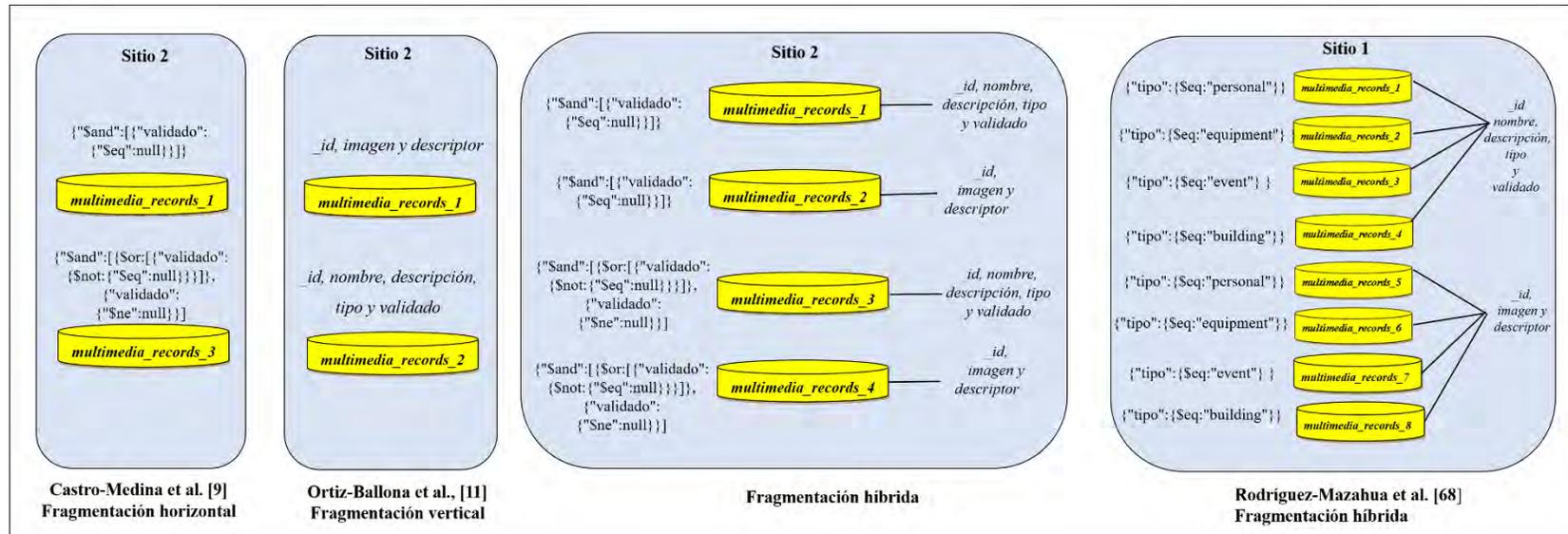


Figura 4.82 Comparación gráfica de la fragmentación híbrida

## Capítulo 5. Conclusiones y recomendaciones

A continuación, se presenta la conclusión con base en el método propuesto en este trabajo de investigación, además, se dan a conocer las recomendaciones como trabajo a futuro para mejorar el método de fragmentación híbrida dinámica para bases de datos multimedia.

### 5.1 Conclusiones

La fragmentación (también conocida como partición) en el área de bases de datos, es una técnica que tiene como objetivo mejorar el desempeño de las consultas y aumentar la disponibilidad de la información al realizar operaciones de inserción, eliminación, actualización y consulta, sobre cualquier tipo de base de datos. Especialmente, las bases de datos multimedia, además de considerar texto, almacenan audio, imagen y video; que al ser de gran tamaño, provocan problemas en la recuperación eficiente de la información.

Como se mencionó al inicio de este trabajo, hay tres enfoques de fragmentación: horizontal, vertical e híbrida o mixta; este último es el resultado de la combinación de las dos anteriores. La mayoría de los trabajos analizados en el estado del arte, relacionados con la partición híbrida abordan la fragmentación estática, es decir, el esquema no se adapta al cambio en los patrones de acceso, solo consideran bases de datos relacionales y no toman en cuenta los datos multimedia. Solo algunos trabajos consideran una fragmentación híbrida dinámica, usando un modelo de costos y este tipo de datos.

El enfoque propuesto y desarrollado en este trabajo fue un método de fragmentación híbrida (horizontal-vertical o vertical-horizontal), que contempla un esquema dinámico, modelo de costos, datos multimedia y un tipo de base de datos diferente a las más comunes (NoSQL).

El modelo de costos propuesto permite comparar esquemas de fragmentación híbrida, con base en la carga de trabajo para posteriormente, obtener y aplicar el esquema menos costoso.

La parte dinámica inicia cuando se ejecuta el vigilante-fragmentador, el cual, permite mejorar la fragmentación híbrida estática, ya que, al ser dinámico, este es capaz de analizar los costos de los fragmentos y de supervisar el umbral de operaciones y desempeño, sin la intervención del DBA, cuando el costo del fragmento supere los umbrales, automáticamente se dispara una nueva fragmentación y los nuevos fragmentos producidos se asignan al sitio correspondiente.

Es importante mencionar que, para lograr implementar la parte dinámica del método de fragmentación híbrida propuesto, es necesario el uso de la aplicación Web XAMANA [9], de acuerdo a los datos ingresados por el administrador de la base de datos (nombre la base de datos, tabla a fragmentar y umbrales (operaciones y desempeño), por mencionar algunos), siendo importantes ambos umbrales para la parte dinámica, XAMANA regresa un esquema de partición híbrida estática y este será la base para realizar la fragmentación híbrida dinámica.

Los resultados obtenidos en este proyecto son relevantes porque se contribuyó al estado del arte, en el área de bases de datos, con un nuevo método de fragmentación híbrida dinámica; siendo la parte dinámica efectiva, puesto que es capaz de adaptarse a las nuevas consultas, manteniendo el buen desempeño de la base de datos a través del tiempo, además de contemplar datos multimedia y un modelo de costos para aplicar el esquema menos costoso.

Finalmente, con esta propuesta, los investigadores tienen una nueva técnica para compararla con sus trabajos, permitiendo nuevas investigaciones en la fragmentación híbrida dinámica. Como trabajo a futuro, este proyecto se adaptará a otros gestores que no sean NoSQL.

## 5.2 Recomendaciones

Al realizar las pruebas del método de fragmentación híbrida dinámica en un entorno real, usando la base de datos HITO, se logró observar las ventajas y desventajas del método propuesto, a continuación, se describen las mejoras como trabajo a futuro.

Recientemente, se incluyeron videos de eventos relevantes en la base de datos HITO, por lo tanto, se recomienda usar la nueva versión de HITO para probar el método desarrollado en este trabajo.

Así mismo, se recomienda contemplar CBIR y CBVR (*Content-Based Video Retrieval*, Recuperación de Video Basada en Contenido) en el método propuesto, además, considerar otros tipos de bases de datos NoSQL, como orientadas a grafos y relacionales.

Finalmente, es necesario realizar un mecanismo que permita al DBA, cuando ya se haya aplicado la fragmentación, deshacer los cambios y obtener la tabla o colección original.

## Productos Académicos



O. Crescencio-Rico, L. Rodríguez-Mazahua, F. Castro-Medina, G. Alor-Hernández, y J. L. Sánchez-Cervantes, «Método de Fragmentación Híbrida Dinámica para Bases de Datos Multimedia», *ICBI*, vol. 11, n.º Especial2, pp. 47-54, sep. 2023.



Oscar Crescencio Rico, Libeth Rodríguez Mazahua, Felipe Castro Medina, Giner Alor Hernández, José Luis Sánchez Cervantes. "MÓDULO DE FRAGMENTACIÓN HÍBRIDA DINÁMICA PARA BASES DE DATOS MULTIMEDIA".

2024

Número de Registro: 03-2024-022010375800-01

**Referencias**

- [1] K. L. Berg, T. Seymour, and R. Goel, “History Of Databases,” *International Journal of Management & Information Systems (IJMIS)*, vol. 17, no. 1, pp. 29–36, Dec. 2012, doi: 10.19030/ijmis.v17i1.7587.
- [2] F. Castro-Medina, L. Rodríguez-Mazahua, A. López-Chau, J. Cervantes, G. Alor-Hernández, and I. Machorro-Cano, “Application of dynamic fragmentation methods in multimedia databases: A review,” *Entropy*, vol. 22, no. 12. MDPI AG, pp. 1–25, Dec. 01, 2020. doi: 10.3390/e22121352.
- [3] M. T. Özsu and P. Valduriez, *Principles of Distributed Database Systems*, Fourth. Cham: Springer International Publishing, 2020. doi: 10.1007/978-3-030-26253-2.
- [4] S. Saad, J. Tekli, R. Chbeir, and K. Yetongnon, “Towards Multimedia Fragmentation,” in *Advances in Databases and Information Systems; Lecture Notes in Computer Science; Manolopoulos, Y., Pokorny, J., Sellis, T.K., Eds*, vol. 4152, Berlin/Heidelberg, Germany: Springer, 2006, pp. 415–429. doi: 10.1007/11827252\_31.
- [5] A. A. Safaei, “Hybrid fragmentation of medical images’ attributes for multidimensional indexing,” *Cluster Comput*, vol. 25, no. 1, pp. 215–230, Feb. 2022, doi: 10.1007/s10586-021-03356-7.
- [6] L. Rodriguez, Xiaou Li, A. D. Cuevas-Rasgado, and F. Garcia-Lamont, “DYVEP: An active database system with vertical partitioning functionality,” in *2013 10th IEEE INTERNATIONAL CONFERENCE ON NETWORKING, SENSING AND CONTROL (ICNSC)*, IEEE, Apr. 2013, pp. 457–462. doi: 10.1109/ICNSC.2013.6548782.
- [7] J. Grant, “Constraint preserving and lossless database transformations,” *Inf Syst*, vol. 9, no. 2, pp. 139–146, Jan. 1984, doi: 10.1016/0306-4379(84)90022-X.
- [8] F. Castro-Medina, L. Rodriguez-Mazahua, A. Lopez-Chau, M. A. Abud-Figueroa, and G. Alor-Hernandez, “FRAGMENT: A Web Application for Database Fragmentation, Allocation and Replication over a Cloud Environment,” *IEEE Latin America Transactions*, vol. 18, no. 06, pp. 1126–1134, Jun. 2020, doi: 10.1109/TLA.2020.9099751.
- [9] F. Castro-Medina *et al.*, “A New Method of Dynamic Horizontal Fragmentation for Multimedia Databases Contemplating Content-Based Queries,” *Electronics (Basel)*, vol. 11, no. 2, p. 288, Jan. 2022, doi: 10.3390/electronics11020288.

- [10] M. Vogt, A. Stiemer, and H. Schuldt, “Polypheny-DB: Towards a Distributed and Self-Adaptive Polystore,” in *2018 IEEE International Conference on Big Data (Big Data)*, IEEE, Dec. 2018, pp. 3364–3373. doi: 10.1109/BigData.2018.8622353.
- [11] A. O. Ortiz Ballona, L. Rodríguez Mazahua, and A. López Chau, “Desarrollo de un método de fragmentación vertical para bases de datos multimedia que considere consultas basadas en contenido,” Tesis de Maestría, Maestría en Sistemas Computacionales, Instituto Tecnológico de Orizaba, Orizaba, 2022. Accessed: Jul. 11, 2022. [Online]. Available: <http://repositorios.orizaba.tecnm.mx:8080/xmlui/handle/123456789/648>
- [12] A. Castillo García, L. Rodríguez Mazahua, and F. Castro-Medina, “Desarrollo de un Método de Fragmentación Horizontal Dinámica para Bases de Datos Multimedia,” Tesis de Maestría, Maestría en Sistemas Computacionales, Instituto Tecnológico de Orizaba, Orizaba, 2022. Accessed: Jul. 11, 2022. [Online]. Available: <http://repositorios.orizaba.tecnm.mx:8080/xmlui/handle/123456789/647>
- [13] A. E. Abdel Raouf, N. L. Badr, and M. F. Tolba, “Distributed Database System (DSS) Design Over a Cloud Environment,” *Multimedia Forensics and Security*, vol. 115, pp. 97–116, Jan. 2017, doi: 10.1007/978-3-319-44270-9\_5.
- [14] F. Castro-Medina, L. Rodríguez-Mazahua, A. López-Chau, G. Alor-Hernández, U. Juárez-Martínez, and C. Sánchez-Ramírez, “An Improvement to FRAGMENT: A Web Application for Database Fragmentation, Allocation, and Replication Over a Cloud Environment,” *Proceedings of Sixth International Congress on Information and Communication Technology.*, 2022, pp. 685–696. doi: 10.1007/978-981-16-1781-2\_60.
- [15] “Apache Lucene - Welcome to Apache Lucene.” Accessed: Dec. 21, 2022. [Online]. Available: <https://lucene.apache.org/>
- [16] G. A. Schreiner, D. Duarte, G. D. Bianco, and R. dos Santos Mello, “A Hybrid Partitioning Strategy for NewSQL Databases: The VoltDB Case,” in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Dec. 2019, pp. 353–360. doi: <https://doi.org/10.1145/3366030.3366062>.
- [17] U. Cetintemel *et al.*, “S-Store: A streaming NewSQL system for big velocity applications,” in *Proceedings of the VLDB Endowment*, Association for Computing Machinery, 2014, pp. 1633–1636. doi: 10.14778/2733004.2733048.

- [18] A. J. Elmore, V. Arora, R. Taft, A. Pavlo, D. Agrawal, and A. el Abbadi, “Squall: Fine-Grained Live Reconfiguration for Partitioned Main Memory Databases,” in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Association for Computing Machinery, May 2015, pp. 299–313. doi: 10.1145/2723372.2723726.
- [19] R. Kallman *et al.*, “H-store: a high-performance, distributed main memory transaction processing system,” *Proceedings of the VLDB Endowment*, vol. 1, no. 2, pp. 1496–1499, Aug. 2008, doi: 10.14778/1454159.1454211.
- [20] A. Pavlo and M. Aslett, “What’s Really New with NewSQL?,” *ACM SIGMOD Record*, vol. 45, no. 2, pp. 45–55, Sep. 2016, doi: 10.1145/3003665.3003674.
- [21] R. Taft *et al.*, “E-store: fine-grained elastic partitioning for distributed transaction processing systems,” *Proceedings of the VLDB Endowment*, vol. 8, no. 3, pp. 245–256, Nov. 2014, doi: 10.14778/2735508.2735514.
- [22] A. Dahal and S. R. Joshi, “A Comparative Analysis on Performance of Minimum Spanning Tree and K-Means Clustering Based Vertical Fragmentation Algorithm,” in *2019 Artificial Intelligence for Transforming Business and Society (AITB)*, IEEE, Nov. 2019, pp. 1–4. doi: 10.1109/AITB48515.2019.8947442.
- [23] A. Noraziah, A. A. C. Fauzi, S. H. S. A. Ubaidillah, B. Alkazemi, and J. B. Odili, “BVAGQ-AR for Fragmented Database Replication Management,” *IEEE Access*, vol. 9, pp. 56168–56177, 2021, doi: 10.1109/ACCESS.2021.3065944.
- [24] Y. F. Ge, M. Orlowska, J. Cao, H. Wang, and Y. Zhang, “Knowledge transfer-based distributed differential evolution for dynamic database fragmentation,” *Knowl Based Syst*, vol. 229, Oct. 2021, doi: 10.1016/j.knosys.2021.107325.
- [25] M. Brendle *et al.*, “SAHARA: Memory Footprint Reduction of Cloud Databases with Automated Table Partitioning,” in *Advances in Database Technology - EDBT*, OpenProceedings.org, 2022, pp. 13–26. doi: 10.5441/002/edbt.2022.02.
- [26] R. Ahmed, B. Nagwa, and T. Mohamed, “Distributed Database System (DSS) Design Over a Cloud Environment,” *Multimedia Forensics and Security*, pp. 97–115, 2017, doi: [https://doi.org/10.1007/978-3-319-44270-9\\_5](https://doi.org/10.1007/978-3-319-44270-9_5).
- [27] M. Aggarwal, S. B. Bajaj, and V. Jaglan, “Vertical Fragmentation using K-Medoid in Distributed Database,” in *Proceedings of the 2022 9th International Conference on Computing for Sustainable Global Development, INDIACom 2022*, Institute of Electrical

- and Electronics Engineers Inc., 2022, pp. 115–118. doi: 10.23919/INDIACom54597.2022.9763186.
- [28] A. Mourão and J. Magalhães, “Towards Cloud Distributed Image Indexing by Sparse Hashing,” in *ICMR 2019 - Proceedings of the 2019 ACM International Conference on Multimedia Retrieval*, Association for Computing Machinery, Inc, Jun. 2019, pp. 288–296. doi: 10.1145/3323873.3325046.
- [29] M. Aharon, M. Elad, and A. Bruckstein, “K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation,” *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006, doi: 10.1109/TSP.2006.881199.
- [30] Y.-F. Ge *et al.*, “Distributed Memetic Algorithm for Outsourced Database Fragmentation,” *IEEE Trans Cybern*, no. 10, pp. 1–14, Oct. 2020, doi: 10.1109/TCYB.2020.3027962.
- [31] J. Jin *et al.*, “Distributed storage system for electric power data based on HBase,” *Big Data Mining and Analytics*, vol. 1, no. 4, pp. 324–334, Dec. 2018, doi: 10.26599/BDMA.2018.9020026.
- [32] “Apache HBase,” Apache. Accessed: Jul. 27, 2022. [Online]. Available: <https://hbase.apache.org/>
- [33] “Apache Hadoop,” Apache. Accessed: Jul. 27, 2022. [Online]. Available: <https://hadoop.apache.org/>
- [34] S. Pan, T. Morris, and U. Adhikari, “Developing a Hybrid Intrusion Detection System Using Data Mining for Power Systems,” *IEEE Trans Smart Grid*, vol. 6, no. 6, pp. 3104–3113, Nov. 2015, doi: 10.1109/TSG.2015.2409775.
- [35] “Common Information Model.” Accessed: Jun. 19, 2022. [Online]. Available: <https://www.dmtf.org/standards/cim>
- [36] A. S. Awad, A. Yousif, and G. Kadoda, “Enhanced Model for Cloud Data Security based on Searchable Encryption and Hybrid Fragmentation,” in *2019 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, IEEE, Sep. 2019, pp. 1–4. doi: 10.1109/ICCCEEE46830.2019.9070918.
- [37] A. Alsirhani, P. Bodorik, and S. Sampalli, “Improving Database Security in Cloud Computing by Fragmentation of Data,” in *2017 International Conference on Computer*

- and Applications (ICCA)*, IEEE, Sep. 2017, pp. 43–49. doi: 10.1109/COMAPP.2017.8079737.
- [38] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, “CryptDB: protecting confidentiality with encrypted query processing,” in *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles - SOSP '11*, New York, New York, USA: ACM Press, 2011, pp. 85–100. doi: 10.1145/2043556.2043566.
- [39] M. Joaquín Rodríguez-Arauz, L. Rodríguez-Mazahua, M. Leoncio Arrijoja-Rodríguez, M. Antonieta Abud-Figueroa, S. Gustavo Peláez-Camarena, and L. del Carmen Martínez-Méndez, *Design of a Multimedia Data Management System that Uses Horizontal Fragmentation to Optimize Content-based Queries*. The Tenth International Conference on Advances in Information Mining and Management, 2020.
- [40] A. Boicea, F. Radulescu, and L. I. Agapin, “MongoDB vs Oracle -- Database Comparison,” in *2012 Third International Conference on Emerging Intelligent Data and Web Technologies*, IEEE, Sep. 2012, pp. 330–335. doi: 10.1109/EIDWT.2012.32.
- [41] JavaServer Faces, “Introducción and JavaServer Faces.” Accessed: Jun. 22, 2022. [Online]. Available: <http://www.jtech.ua.es/>
- [42] NetBeans, “NetBeans Docs & support.” Accessed: Jun. 22, 2022. [Online]. Available: <https://netbeans.apache.org/kb/index.html>
- [43] C. Nieves-Guerrero, J. Ucán-Pech, and V. Menéndez-Domínguez, “UWE en Sistema de Recomendación de Objetos de Aprendizaje. Aplicando Ingeniería Web: Un Método en Caso de Estudio,” *Revista Latinoamericana de Ingeniería de Software*, vol. 2, no. 3, p. 137, Jun. 2014, doi: 10.18294/relais.2014.137-143.
- [44] Boofcv.org, “BoofCV.” Accessed: Jun. 22, 2022. [Online]. Available: [https://boofcv.org/index.php?title=Main\\_Page](https://boofcv.org/index.php?title=Main_Page)
- [45] K. Fasolin *et al.*, “Efficient Execution of Conjunctive Complex Queries on Big Multimedia Databases,” in *2013 IEEE International Symposium on Multimedia*, IEEE, Dec. 2013, pp. 536–543. doi: 10.1109/ISM.2013.112.
- [46] H. Jégou, M. Douze, and C. Schmid, “Product Quantization for Nearest Neighbor Search,” *IEEE Trans Pattern Anal Mach Intell*, vol. 33, no. 1, pp. 117–128, Jan. 2011, doi: 10.1109/TPAMI.2010.57.

- [47] H. Jégou, R. Tavenard, M. Douze, and L. Amsaleg, “Searching in one billion vectors: re-rank with source coding,” Feb. 2011, [Online]. Available: <http://arxiv.org/abs/1102.3828>
- [48] “What is Java and why do I need it?” Accessed: Mar. 20, 2023. [Online]. Available: [https://www.java.com/en/download/help/whatis\\_java.html](https://www.java.com/en/download/help/whatis_java.html)
- [49] “JavaServer Faces Technology,” Oracle. Accessed: Mar. 20, 2023. [Online]. Available: <https://www.oracle.com/java/technologies/javaserverfaces.html>
- [50] “Introducción a JavaServer Faces,” Dept. Ciencia de la Computación e IA Universidad de Alicante. Accessed: Mar. 20, 2023. [Online]. Available: <http://www.jtech.ua.es/j2ee/publico/jsf-2012-13/sesion01-apuntes.html>
- [51] “NetBeans IDE ,” Oracle. Accessed: Jul. 21, 2022. [Online]. Available: <https://www.oracle.com/mx/tools/technologies/netbeans-ide.html>
- [52] *Unified Modeling Language*. Object Management Group. Accessed: Jul. 17, 2022. [Online]. Available: <https://www.uml.org/>
- [53] J. Molina-Ríos and N. Pedreira-Souto, “Comparison of development methodologies in web applications,” *Information and Software Technology*, vol. 119. Elsevier B.V., Mar. 01, 2020. doi: 10.1016/j.infsof.2019.106238.
- [54] N. Koch, A. Knapp, G. Zhang, and H. Baumeister, “Uml-Based Web Engineering,” in *Web Engineering: Modelling and Implementing Web Applications*, London: Springer London, 2008, pp. 157–191. doi: 10.1007/978-1-84628-923-1\_7.
- [55] “What is MySQL?,” Oracle. Accessed: Jul. 30, 2022. [Online]. Available: <https://www.oracle.com/mysql/what-is-mysql/>
- [56] “What Is MongoDB?,” MongoDB. Accessed: Jul. 21, 2022. [Online]. Available: <https://www.mongodb.com/es/what-is-mongodb>
- [57] R. Chbeir and D. Laurent, “Enhancing Multimedia Data Fragmentation,” *Journal of Multimedia Processing and Technologies*, vol. 1, no. 2, pp. 112–130, 2010.
- [58] P. Kling, M. T. Özsu, and K. Daudjee, “Scaling XML query processing: Distribution, localization and pruning,” *Distrib Parallel Databases*, vol. 29, no. 5–6, pp. 445–490, Oct. 2011, doi: 10.1007/s10619-011-7085-8.
- [59] A. Jindal and J. Dittrich, “Relax and Let the Database Do the Partitioning Online,” in *Enabling Real-Time Business Intelligence*, vol. 126, M. Castellanos, U. Dayal, and W.

- Lehner, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 65–80. doi: [https://doi.org/10.1007/978-3-642-33500-6\\_5](https://doi.org/10.1007/978-3-642-33500-6_5).
- [60] N. Gorla, V. Ng, and D. M. Law, “Improving database performance with a mixed fragmentation design,” *J Intell Inf Syst*, vol. 39, no. 3, pp. 559–576, Dec. 2012, doi: [10.1007/s10844-012-0203-x](https://doi.org/10.1007/s10844-012-0203-x).
- [61] S. Song and L. Chen, “Indexing dataspace with partitions,” *World Wide Web*, vol. 16, no. 2, pp. 141–170, Mar. 2013, doi: [10.1007/s11280-012-0163-7](https://doi.org/10.1007/s11280-012-0163-7).
- [62] S. Chen, C. H. Chi, C. Ding, and R. K. Wong, “Data decomposition based partial replication model for software services,” in *Proceedings - IEEE 10th International Conference on Services Computing, SCC 2013*, 2013, pp. 256–263. doi: [10.1109/SCC.2013.83](https://doi.org/10.1109/SCC.2013.83).
- [63] X. Wang, X. Fan, J. Chen, and X. Du, “Automatic Data Distribution in Large-scale OLTP Applications,” *International Journal of Database Theory and Application*, vol. 7, no. 4, pp. 37–46, Aug. 2014, doi: [10.14257/ijdta.2014.7.4.04](https://doi.org/10.14257/ijdta.2014.7.4.04).
- [64] M. Kechar and S. N. Bahloul, “Hybrid Fragmentation of XML Data Warehouse Using K-Means Algorithm,” in *Advances in Databases and Information Systems*, Y. Manolopoulos, G. Trajcevski, and M. Kon-Popovska, Eds., Springer International Publishing, 2014, pp. 70–82. doi: [10.1007/978-3-319-10933-6\\_6](https://doi.org/10.1007/978-3-319-10933-6_6).
- [65] K. Chen, Y. Zhou, and Y. Cao, “Online data partitioning in distributed database systems,” in *EDBT 2015 - 18th International Conference on Extending Database Technology, Proceedings*, OpenProceedings.org, University of Konstanz, University Library, 2015, pp. 1–12. doi: [10.5441/002/edbt.2015.02](https://doi.org/10.5441/002/edbt.2015.02).
- [66] S. Harikumar and R. Ramachandran, “Hybridized fragmentation of very large databases using clustering,” in *2015 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems, SPICES 2015*, Institute of Electrical and Electronics Engineers Inc., Apr. 2015. doi: [10.1109/SPICES.2015.7091488](https://doi.org/10.1109/SPICES.2015.7091488).
- [67] M. Al-Kateb, P. Sinclair, G. Au, and C. Ballinger, “Hybrid row-column partitioning in teradata ®,” *Proceedings of the VLDB Endowment*, vol. 9, no. 13, pp. 1353–1364, Sep. 2016, doi: [10.14778/3007263.3007273](https://doi.org/10.14778/3007263.3007273).
- [68] L. Rodríguez-Mazahua, G. Alor-Hernández, J. Cervantes, A. López-Chau, and J. L. Sánchez-Cervantes, “A hybrid partitioning method for multimedia databases,” *Dyna (Medellin)*, vol. 83, no. 198, p. 59, Sep. 2016, doi: [10.15446/dyna.v83n198.50507](https://doi.org/10.15446/dyna.v83n198.50507).

- [69] T. Padiya, J. J. Kanwar, and M. Bhise, “Workload aware hybrid partitioning,” in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Oct. 2016, pp. 51–58. doi: 10.1145/2998476.2998479.
- [70] L. Sun, M. J. Franklin, J. Wang, and E. Wu, “Skipping-oriented partitioning for columnar layouts,” *Proceedings of the VLDB Endowment*, vol. 10, no. 4, pp. 421–432, Nov. 2016, doi: 10.14778/3025111.3025123.
- [71] S. Rani, D. K. Koshley, and R. Halder, “Partitioning-insensitive watermarking approach for distributed relational databases,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Verlag, 2017, pp. 172–192. doi: 10.1007/978-3-662-56266-6\_8.
- [72] A. Mourão and J. Magalhães, “Balancing search space partitions by sparse coding for distributed redundant media indexing and retrieval,” *Int J Multimed Inf Retr*, vol. 7, no. 1, pp. 57–70, Mar. 2018, doi: 10.1007/s13735-017-0140-0.
- [73] G. C. Durand *et al.*, “GridFormation: Towards Self-Driven Online Data Partitioning using Reinforcement Learning,” in *Proceedings of the First International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*, New York, NY, USA: ACM, Jun. 2018, pp. 1–7. doi: 10.1145/3211954.3211956.
- [74] K.-L. Koong, S.-C. Haw, and L.-K. Soon, “Labeling-Based Hybrid XML Fragmentation Model,” *Adv Sci Lett*, vol. 24, no. 2, pp. 1177–1181, Mar. 2018, doi: 10.1166/asl.2018.10711.
- [75] G. A. Schreiner, D. Duarte, O. : Ronaldo, and S. Mello, “An autonomous hybrid data partition for NewSQL DBs,” in *SBB D Companion*, 2018, pp. 95–101.
- [76] M. Vogt, A. Stiemer, and H. Schuldt, “Polypheny-DB: Towards a Distributed and Self-Adaptive Polystore,” in *2018 IEEE International Conference on Big Data (Big Data)*, IEEE, Dec. 2018, pp. 3364–3373. doi: 10.1109/BigData.2018.8622353.
- [77] T. Chawla, G. Singh, and E. S. Pilli, “HYPSO: Hybrid partitioning for big RDF storage and query processing,” in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Jan. 2019, pp. 188–194. doi: 10.1145/3297001.3297025.
- [78] S. Badran, N. Arman, and M. Farajallah, “Towards a hybrid data partitioning technique for secure data outsourcing,” in *Proceedings - 2020 21st International Arab Conference on Information Technology, ACIT 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020. doi: 10.1109/ACIT50332.2020.9300064.

- [79] M. Pinnecke, G. Campero Durand, D. Broneske, R. Zoun, and G. Saake, “GridTables: A One-Size-Fits-Most H2TAP Data Store,” *Datenbank-Spektrum*, vol. 20, no. 1, pp. 43–56, Mar. 2020, doi: 10.1007/s13222-019-00330-x.
- [80] V. Kulba and S. Somov, “Dynamic Fragment Allocation in Distributed System with Time-Varying Parameters of its Operation,” in *2020 13th International Conference “Management of large-scale system development” (MLSD)*, IEEE, Sep. 2020, pp. 1–4. doi: 10.1109/MLSD49919.2020.9247671.
- [81] D. Kang, R. Jiang, and S. Blanas, “Jigsaw: A Data Storage and Query Processing Engine for Irregular Table Partitioning,” in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Association for Computing Machinery, 2021, pp. 898–911. doi: 10.1145/3448016.3457547.
- [82] S. Badran, N. Arman, and M. Farajallah, “An Efficient Approach for Secure Data Outsourcing using Hybrid Data Partitioning,” in *2021 International Conference on Information Technology, ICIT 2021 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., Jul. 2021, pp. 418–423. doi: 10.1109/ICIT52682.2021.9491745.
- [83] A. Azila, C. Fauzi, W. Fariza, W. A. Rahman, A. Fauzi, and F. Weigelt, “Managing Fragmented Database in Distributed Database Environment,” *Journal of Mathematics and Computing Science*, vol. 7, no. 1, pp. 8–14, 2021.
- [84] M. Patel, N. Yadav, and M. Bhise, “Workload Aware Cost-Based Partial Loading of Raw Data for Limited Storage Resources,” in *Futuristic Trends in Networks and Computing Technologies*, vol. 936, Springer, 2022, pp. 1035–1048. doi: 10.1007/978-981-19-5037-7\_74.
- [85] Z. J. Ahmed and S. T. Alluhaibi, “Hybrid Data Fragmentation Using Genetic Killer Whale Optimization-Based Clustering Model,” *J Pharm Negat Results*, vol. 13, no. SO2, Jan. 2022, doi: 10.47750/pnr.2022.13.s02.39.
- [86] R. Cantini *et al.*, “Block size estimation for data partitioning in HPC applications using machine learning techniques,” Nov. 2022, [Online]. Available: <http://arxiv.org/abs/2211.10819>