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 vertical para bases de datos multimedia que considere consultas basadas en contenido"

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN SISTEMAS COMPUTACIONALES

PRESENTA:

I.S.C. Aldo Osmar Ortiz Ballona

DIRECTOR DE TESIS:

Dra. Lisbeth Rodríguez Mazahua

CODIRECTOR DE TESIS:

Dr. Asdrúbal López Chau

ORIZABA, VERACRUZ, MÉXICO. MAYO 2022

Orizaba, Veracruz, 18/mayo/2022
Dependencia: **División de Estudios de Posgrado e Investigación**
Asunto: **Autorización de Impresión**
OPCION: **I**

C. ALDO OSMAR ORTIZ BALLONA
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 vertical para bases de datos multimedia que considere consultas basadas en contenido"

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

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
CIENCIA – TÉCNICA - CULTURA®

EDUCACIÓN | TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE ORIZABA
DIV. DE EST. DE POSGRADO E INVEST.

DR. MARIO LEONCIO ARRIOJA RODRÍGUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN

OG-13-F06

Orizaba, Veracruz, 01/marzo/2022
Asunto: **Revisión de trabajo escrito**

C. MARIO LEONCIO ARRIOJA RODRÍGUEZ
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.

ALDO OSMAR ORTIZ BALLONA

la cual lleva el título de:

"Desarrollo de un método de fragmentación vertical para bases de datos multimedia que considere consultas basadas en contenido"

y concluyen que se acepta.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
CIENCIA – TÉCNICA - CULTURA®

PRESIDENTE:	DRA. LISBETH RODRÍGUEZ MAZAHUA	FIRMA
SECRETARIO:	M.C. MA. ANTONIETA ABUD FIGUEROA	FIRMA
VOCAL:	M.C. CELIA ROMERO TORRES	FIRMA
VOCAL SUP.:	DR. ASDRÚBAL LÓPEZ CHAU	FIRMA

TA-09-21

Agradecimientos

Quiero agradecer a mis padres, hermanos, amigos, tíos y a todas aquellas personas que ofrecieron palabras de aliento y comprendían mi situación aportando desde un abrazo hasta apoyo económico que fue suficiente para continuar con esta meta.

Gracias a mis compañeros y amigos que siempre estuvieron para explicar algo que no comprendía, que desde que iniciamos, no dejamos de ser buenos compañeros y espero que perdure el resto de nuestras vidas.

Quiero agradecer a mis amigos, Juan y Oscar, por todo el apoyo ofrecido y haber estado desde el principio en este proceso.

A Eduardo y Daniel, gracias por el apoyo y las palabras de ánimos.

Doy gracias a la Doctora Lisbeth por aceptarme como tesista, por su paciencia y apoyo para terminar el trabajo de investigación.

Agradezco a la Maestra María Antonieta y a la Maestra Beatriz por ser excelentes docentes y brindarme la oportunidad de continuar creciendo académicamente.

Gracias al Maestro Luis por la amistad, pláticas y consejos.

Gracias a la maestra Celia por la facilidad ofrecida para los trámites académicos y de la beca.

Agradezco al Consejo Nacional de Ciencia y Tecnología (CONACyT) por el apoyo de la beca de manutención otorgada durante el periodo de estudios y al Tecnológico Nacional de México (TecNM) por dar soporte a esta investigación.

Índice

Índice de figuras

Índice de tablas

Resumen

La fragmentación de datos provee de ventajas en la recuperación, disponibilidad y el desempeño de bases de datos. Esta técnica se utiliza en bases de datos multimedia para reducir el costo de ejecución de las consultas. En este tipo de bases de datos, las consultas basadas en contenido son ampliamente realizadas.

El objetivo del presente trabajo es desarrollar un método de fragmentación vertical para base de datos multimedia que optimice consultas basadas en contenido, mediante un análisis comparativo de las técnicas propuestas en la literatura en los últimos diez años se seleccionó la más adecuada que se ocupó para evaluar el método desarrollado. La contribución es una técnica efectiva para la fragmentación vertical que se integró a una aplicación web para fragmentación dinámica de base de datos multimedia.

Las tecnologías empleadas son el lenguaje de programación Java, JavaServer Faces como marco de trabajo, el gestor de bases de datos Postgres-XL y NetBeans como IDE (*Integrated Develoment Environment*, entorno de desarrollo integrado), ya que este es el entorno que mejor se adapta para el desarrollo con estas tecnologías.

Abstract

Data fragmentation provides advantages in database recovery, availability, and performance. This technique is used in multimedia databases to reduce the cost of executing queries. In this type of database, content-based queries are widely performed.

The objective of this work is to develop a vertical fragmentation method for multimedia databases that optimizes content-based queries. Through a comparative analysis of the techniques proposed in the literature in the last ten years, the most appropriate one that was used to evaluate the method developed. Contribution is an effective technique for vertical fragmentation that was integrated into a web application for dynamic multimedia database fragmentation.

The technologies used are the Java programming language, Java Server Faces as the framework, the Postgres-XL database management system, and NetBeans as IDE (Integrated Development Environment), since this is the environment that best adapts for development with these technologies

Introducción

El presente trabajo tiene como objetivo desarrollar el módulo encargado de fragmentar bases de datos multimedia de manera vertical considerando fragmentos que mantengan unidos los datos multimedia y los atributos que contengan los datos que los describen, también, divide el resto de los atributos tradicionales en fragmentos colocados en sitios donde fueron más requeridos mediante el cálculo de costos por atributos.

Aplicar métodos de fragmentación a sistemas de bases de datos multimedia es esencial, ya que actualmente la demanda de información es enorme y contribuyen a la disponibilidad de los datos y a la fiabilidad de estos sistemas. La mayoría de los métodos propuestos en la literatura se enfocan en fragmentación horizontal, que consiste en dividir una tabla de la base de datos en subconjuntos de tuplas.

En esta investigación se desarrolló un nuevo método de fragmentación vertical, ya que los pocos que existen no optimizan consultas basadas en contenido como rango o *k* vecinos más cercanos, las cuales son muy utilizadas en bases de datos multimedia. Este trabajo resuelve la problemática observada haciendo primero un análisis comparativo de las técnicas de fragmentación vertical existentes, para conocer sus ventajas y desventajas y con base en esto diseñar un nuevo método para su posterior implementación y validación. De esta manera se obtiene un método de fragmentación vertical que permite reducir el costo de ejecución de consultas basadas en contenido.

En el primer capítulo de este trabajo se da a conocer los conceptos fundamentales utilizados a lo largo del proyecto, además se muestran los objetivos, el planteamiento del problema y la justificación. En el segundo capítulo se realiza un análisis de diferentes artículos relacionados con la fragmentación vertical y sistemas CBIR (*Content-Based Image Retrieval*, Recuperación de Imágenes Basadas en Contenido) para entender el funcionamiento. El tercer capítulo presenta el desarrollo del método siguiendo la metodología UWE; mientras que como evaluación se utilizó un caso de estudio y una comparación con el método elegido en el análisis del capítulo anterior.

Finalmente, en el capítulo cinco se muestran los resultados y conclusiones de la tesis, así como recomendaciones para mejorar el trabajo en un futuro.

Capítulo 1. Antecedentes

1.1 Marco teórico

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

1.1.1 Base de datos multimedia

Una base de datos multimedia se refiere a una colección de datos en la que existen múltiples modalidades de datos, como texto e imágenes. En este sistema de base de datos, la información de las diferentes modalidades está relacionada entre sí. Por ejemplo, los datos de texto están relacionados con las imágenes como su información de anotación [1].

1.1.2 Consultas basadas en contenido

Los sistemas CBIR (*Content-Based Image Retrieval*, Recuperación de imágenes basada en contenido) emplean un conjunto de técnicas para gestionar imágenes digitales en función de su contenido visual. Para representar los datos, un sistema CBIR envía las imágenes a un extractor de características, que genera un vector de datos que representa la información específica del contenido visual de las imágenes. Los ejemplos incluyen extractores de características basados en color, textura y forma [2].

1.1.2.1 Consultas rango

Se representan por Rq(S_q,ξ), donde S_q es el vector de características del elemento de consulta y ξ es el radio; recuperan todos los elementos a una distancia de ξ desde S_q. Este tipo de consulta se aplica a escenarios específicos, donde un especialista conoce el valor adecuado de ξ para el dominio de datos, considerando los extractores de características y las funciones de distancia que se utilizan [2].

1.1.2.1 Consultas *k* vecinos más cercanos

Representado como *k*NNq(S_q,k), donde S_q es el vector de característica del elemento de consulta y k es el número de elementos a ser devueltos, recupera los k elementos más similares a S_q usualmente clasificados desde el más cercano (1er elemento) hasta el más lejano (k-elemento) [2].

1.1.3 Fragmentación

La fragmentación es el proceso de dividir las tablas relacionales ya sea a través de un operador de selección o de proyección, dicho proceso puede anidar más procesos de fragmentación [3]. Fragmentar una base de datos permite reducir el tiempo de respuesta de las consultas y disminuir su costo de ejecución.

1.1.4 Tipos de fragmentación

En la fragmentación existen enfoques que se utilizan dependiendo de las necesidades para la solución de problemas particulares, como la fragmentación vertical o fragmentación horizontal que a continuación se describen en este documento.

1.1.4.1 Fragmentación horizontal

La fragmentación horizontal divide una relación a lo largo de sus tuplas. Así, cada fragmento tiene un subconjunto de las tuplas de la relación. Hay dos versiones de partición horizontal: primaria y derivada. La fragmentación horizontal primaria de una relación se realiza utilizando predicados que se definen sobre esa relación. La fragmentación horizontal derivada, por otra parte, es la partición de una relación que resulta de predicados definidos en otra relación [3].

1.1.4.1.1 Fragmentación horizontal (primaria)

La información de la base de datos que se requiere en este tipo de fragmentación se refiere al esquema conceptual global, principalmente sobre cómo se conectan las relaciones entre sí, especialmente con las uniones. Una forma de captar esta información es modelar explícitamente las relaciones de unión de claves primarias y externas en un grafo de unión. En este grafo, cada relación R_i se representa como un vértice y un eje dirigido L_k existe de R_i a R_j sí hay un *equijoin* de clave primaria de R_i a R_j. L_k también representa una relación de uno a muchos.

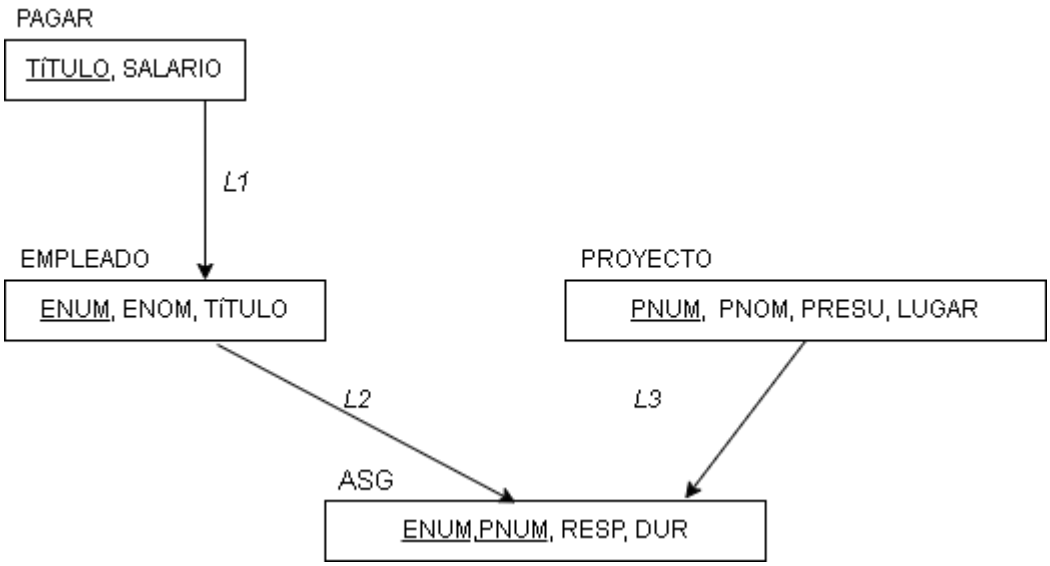


Figura 1.1 Grafo de unión que representa relaciones entre relaciones

La Figura 1.1 muestra los ejes entre las relaciones de una base de datos. Se observa que la dirección del eje muestra una relación de uno a muchos. Por ejemplo, para cada título hay varios empleados con ese título; por lo tanto, hay un eje entre las relaciones PAGAR y EMPLEADO. De la misma forma, la relación de muchos a muchos entre las relaciones EMPLEADO y PROYECTO se expresa con dos ejes a la relación ASG. La relación desde donde surge el eje se denomina la fuente del eje y la relación en la cabeza del eje es la de destino.

La fragmentación horizontal primaria se aplica a las relaciones que no tienen ejes entrantes en el grafo de unión y se realiza utilizando los predicados que se definen en esa relación.

Una fragmentación horizontal primaria se define por una operación de selección en las relaciones de fuente de un esquema de base de datos. Por lo tanto, dada la relación R sus fragmentos horizontales están dados por:

$$R_i = \sigma F_i(\mathrm{R}),\ 1 \leq i \leq w$$

donde F_i es la fórmula de selección usada para obtener el fragmento R_i (también llamado predicado de fragmentación). Si F_i está en forma normal conjuntiva, es un predicado mintérmino (*mi*). Las consultas de usuario a menudo incluyen predicados más complicados, que son combinaciones booleanas de predicados simples. Una de esas combinaciones, llamada predicado mintérmino, es la conjunción de predicados simples. Dado que siempre es posible transformar una expresión booleana en una forma normal conjuntiva, el uso de predicados mintérmino en los algoritmos de diseño no causa ninguna pérdida de generalidad [3].

1.1.4.1.2 Fragmentación horizontal (derivada)

Una fragmentación horizontal derivada se aplica a las relaciones de destino en el grafo de unión y se realiza con base en predicados definidos sobre la relación de origen del eje del grafo de unión [3].

1.1.4.2 Fragmentación vertical

La partición vertical es intrínsecamente más complicada que la horizontal, principalmente debido al número total de alternativas posibles. Por ejemplo, en la partición horizontal, si el número total de predicados simples es *n*, hay 2^n predicados mintérmino posibles. Además, algunos de ellos contradirán las implicaciones existentes, reduciendo aún más los fragmentos candidatos que deben considerarse.

Sin embargo, en el caso de la partición vertical, si una relación tiene *m* atributos que no formen la clave primaria, el número de fragmentos posibles es igual a B(*m*), que es el número *m*-ésimo de Bell. Para valores grandes de *m*, B(*m*) $\approx m^m$; por ejemplo, para *m* = 10, B(*m*) ≈ 115, 000, para *m* = 15, B(*m*) $\approx 10^9$, para *m* = 30, B(*m*) = 10^{23}.

Estos valores indican que es inútil intentar obtener soluciones óptimas para el problema de partición vertical; hay que recurrir a la heurística. Existen dos tipos de enfoques heurísticos para la fragmentación vertical de las relaciones globales:

- Agrupación: comienza asignando cada atributo a un fragmento y, en cada paso, une algunos de los fragmentos hasta que se satisfacen algunos criterios.

- Partición: comienza con una relación y decide sobre particiones beneficiosas basadas en el comportamiento de acceso de las aplicaciones a los atributos [3].

1.1.4.3 Fragmentación híbrida

Una simple fragmentación horizontal o vertical de un esquema de base de datos puede no ser suficiente para satisfacer los requisitos de las aplicaciones de los usuarios.

La fragmentación híbrida se presenta cuando una fragmentación vertical puede ir seguida de una horizontal, o viceversa, produciendo un árbol estructurado de particiones, dado que los dos tipos de estrategias de partición se aplican una tras otra, esta alternativa se llama fragmentación híbrida, fragmentación mixta o fragmentación anidada [3].

1.2 Situación tecnológica, económica y operativa de la empresa

El Instituto Tecnológico de Orizaba (ITO) es una institución que pertenece al Tecnológico Nacional de México, que se encuentra ubicado en Oriente 9, Colonia Emiliano Zapata, en la ciudad de Orizaba, Veracruz. Esta institución ofrece carreras de licenciatura, maestría y doctorado. En el área de maestrías ofrece las carreras de Maestría en Ingeniería Electrónica, Maestría en Ingeniería Industrial, Maestría en Ciencias en Ingeniería Química, Maestría en Ingeniería Administrativa y la Maestría en Sistemas Computacionales.

1.3 Planteamiento del problema

En los años pasados, las aplicaciones multimedia distribuidas se han vuelto cada vez más populares; como resultado, las técnicas de fragmentación se han adaptado a un contexto multimedia para lograr apropiadamente una utilización alta de recursos e incrementar la concurrencia y el paralelismo [4]. Aunque la mayoría de las propuestas presentadas en la literatura se enfocan en fragmentación horizontal [4]–[8]. La fragmentación vertical se reconoce como una técnica adecuada para mejorar el desempeño de las consultas en bases de datos multimedia [9], [10], [11]. Una técnica de fragmentación vertical se aplicó en un sistema *e-learning* de base de datos de videos para lograr la ejecución eficiente de consultas [12]. La desventaja de este método

es que no consideraba ni los costos de transporte de los objetos multimedia a través de los nodos de la red, ni el tamaño de los objetos multimedia. En investigaciones recientes se eliminó dicha desventaja [13]–[16]. Sin embargo, los métodos propuestos no toman en cuenta consultas basadas en contenido. Este tipo de consultas son necesarias cuando las anotaciones textuales son inexistentes o incompletas. Además, los métodos basados en contenido mejoran potencialmente la recuperación incluso cuando las anotaciones textuales están presentes, ya que dan un conocimiento adicional a las colecciones de datos multimedia [17]. La mayoría de las propuestas relacionadas con fragmentación vertical en la literatura no consideran datos multimedia [18]–[20]. Por otro lado, las propuestas que toman en cuenta datos multimedia no optimizan consultas basadas en contenido [12], [13], [14], [16], [21]. En este proyecto se desarrolló un método de fragmentación vertical para bases de datos multimedia que toma en cuenta consultas basadas en contenido.

1.4 Objetivos generales y específicos

A continuación, se muestra el objetivo general y específicos.

1.4.1 Objetivo general

Desarrollar un método de fragmentación vertical para bases de datos multimedia que permita reducir el tiempo de respuesta y el costo de ejecución de consultas basadas en contenido.

1.4.2 Objetivo general

1 Estudiar y analizar el estado del arte de los métodos de fragmentación vertical para bases de datos multimedia, así como de los modelos de costo utilizados para evaluar esquemas de fragmentación.

2 Realizar un análisis comparativo de los algoritmos y de los modelos de costo identificados en el estado del arte para conocer sus ventajas y desventajas, además de determinar 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 costo para la fragmentación vertical de bases de datos multimedia que consideren consultas basadas en contenido.

5 Implementar el método y modelo de costo utilizando las tecnologías seleccionadas.

6 Comparar los métodos y modelos de costo con un algoritmo y modelo seleccionados del estado del arte.

1.5 Justificación

Debido a las necesidades presentadas se requiere mostrar en este trabajo un enfoque de fragmentación vertical para bases de datos multimedia, que considere consultas basadas en contenido, esto permite un mejor rendimiento en estas bases de datos reduciendo el costo de ejecución y el tiempo de respuesta de las consultas basadas en contenido. Por medio de este trabajo se benefician los investigadores del área de base de datos, ya que cuentan con un método que utilizarán para obtener esquemas de fragmentación vertical óptimos y/o para compararlo con las técnicas desarrolladas. Además, el método propuesto se integró en una aplicación Web para la fragmentación dinámica de bases de datos multimedia que se está desarrollando en una tesis del Doctorado en Ciencias de la Ingeniería de este Instituto.

Capítulo 2. Estado de la práctica

Antes de introducirse al tema por completo, se revisó un considerable número de artículos de los cuales se tomó la información más relevante y se analizaron meticulosamente las investigaciones relacionadas con el tema propuesto.

A continuación, se presenta una breve descripción de los más importantes.

2.1 Trabajos relacionados

En [13] los autores presentaron un algoritmo de fragmentación vertical para bases de datos multimedia distribuidas (MAVP, *Multimedia Adaptable Vertical Partitioning*, Fragmentación Vertical Multimedia Adaptable) que toma en cuenta el tamaño de los objetos multimedia para generar un esquema de fragmentación vertical óptimo. MAVP minimiza la cantidad de accesos a datos irrelevantes y el costo de transporte de las consultas en bases de datos multimedia distribuidas para lograr una recuperación eficiente de objetos multimedia. MAVP requiere como entrada la matriz de uso de atributos y el tamaño de estos. El algoritmo soporta la fragmentación vertical de *n* maneras y la fragmentación vertical con el mejor ajuste. La primera genera el número específico de fragmentos requeridos por el usuario y la segunda genera una partición óptima general que minimiza el costo de procesamiento de consultas sin restricción en el número de fragmentos generados. En este trabajo se presentó el desarrollo de un modelo de costos el cual considera el costo de procesamiento general de consultas en un ambiente distribuido multimedia. Como trabajo a futuro se mencionó la fragmentación vertical dinámica en bases de datos distribuidas multimedia basada en los cambios de las consultas.

Existen métodos y algoritmos que intentan dar una solución óptima en la fragmentación de bases de datos, los sistemas de bases de datos distribuidas se apoyan de estos métodos para lograr un buen rendimiento, estos sistemas son los encargados de administrar el correcto funcionamiento de una base de datos distribuida, fragmentan y distribuyen los fragmentos de tal manera que sea

transparente para los usuarios y exista un bajo costo de transmisión de datos cuando se realice alguna operación. En [18] los autores presentaron un algoritmo híbrido de los algoritmos *Differential Evolution* (DE, algoritmo de evolución diferencial) y BEA (*Bond Energy Algorithm*, algoritmo de energía de enlace), los autores se centraron en mejorar la calidad de fragmentos generados por los algoritmos mediante los términos de GAM (*global affinity measure*, medida de afinidad global), consideraron 11 conjuntos de datos de muestra, utilizados para crear una matriz de afinidad de atributos. Programaron el algoritmo en Python 3, en una computadora equipada con un procesador Intel Core i5 a 1.6 GHz, 8 Gb de RAM y con el sistema operativo Mac OS High Sierra. Obtuvieron resultados favorables para la fragmentación vertical, dando un mejor rendimiento, ya que consideró un mayor número de posibles soluciones en comparación al algoritmo de energía de enlace clásico.

Rahimi, Parand y Riahi [19] propusieron resolver el problema que trae consigo la fragmentación vertical de bases de datos y la asignación de los fragmentos por separado, fusionar los métodos y realizar simultáneamente las dos tareas para la mejora del rendimiento del sistema de base de datos distribuida. Para lograr su objetivo, aplicaron *Bond Energy Algorithm* (BEA) con una medida de afinidad modificada en un proceso jerárquico y calcularon el costo de asignación de datos para cada sitio y así colocar el fragmento en el sitio adecuado.

Hadoop se ha convertido en una arquitectura líder para el procesamiento de datos a gran escala. Una de las formas eficientes de acelerar el procesamiento de datos es la técnica de almacenamiento orientada a columnas que se ha integrado recientemente en la familia Hadoop. Sin embargo, diseñar un algoritmo de agrupación de atributos apropiado para lograr un rendimiento óptimo de procesamiento de datos en el entorno de Hadoop orientado a columnas fue un gran obstáculo. En [22] los autores propusieron un nuevo algoritmo llamado *Column-oriented Hadoop based Attribute Clustering* (CHAC) para resolver este problema. Ambos casos de agrupación de atributos traslapados y no traslapados se consideran en CHAC. Además, también se tuvo en cuenta un parámetro ajustable para prohibir la redundancia excesiva de atributos limitando la sobrecarga de espacio. Para realizar las pruebas se utilizó el *benchmark* TPC-H y el algoritmo se evalúo en 16

nodos de los cuales uno de ellos fungió como maestro. La base de datos contuvo 30 atributos y 20 GB de tamaño. Se observó que los resultados generados por el modelo de costos están estrechamente relacionados con el tiempo de ejecución de las consultas en las fases del mapeo, ya que su tendencia es consistente, lo que indica la efectividad del modelo de costos propuesto.

Las bases de datos tradicionales no están equipadas con la funcionalidad adecuada para manejar el volumen y la variedad de Big Data. Zhao, Cheng y Rusu [23] investigaron el problema del procesamiento de datos brutos con base en consultas con carga parcial de atributos. Se modeló la carga como fragmentación vertical binaria completamente replicada. Se proporcionó una formulación lineal de optimización de programación de enteros mixtos que demostró ser *NP-hard*. Se diseñó una heurística con dos etapas que encuentra una solución cercana a la solución óptima en una fracción del tiempo. Se extendió la formulación de optimización y la heurística para el procesamiento lineal de datos brutos, escenario en el que el acceso y la extracción de datos se ejecutan simultáneamente. Se proporcionaron tres casos de estudio sobre formatos de datos reales que confirmaron la precisión del modelo cuando es implementado en un operador lineal obtenido del estado del arte para el procesamiento de datos brutos.

Encontrar el esquema de partición vertical adecuado para una carga de trabajo es uno de los problemas esenciales de optimización de la base de datos. Con la partición adecuada, las consultas y las tareas de administración pueden omitir datos innecesarios, mejorando su rendimiento. En [24] los autores consideraron la viabilidad de una solución general de aprendizaje automático para superar los inconvenientes de los enfoques más comunes. Se amplió el trabajo en *GridFormation*, asignando la tarea de partición a una tarea de RL (*Reinforcement learning*, Aprendizaje de Refuerzo). Se validó la propuesta experimentalmente utilizando una base de datos y una carga de trabajo con el *benchmark* TPC-H y el marco de trabajo de Google Dopamine para RL profundo. Se presentaron tiempos de ejecución competitivos mientras aumenta el número de atributos en una tabla, superando a algunos algoritmos de vanguardia.

Costa, Costa y Santos [25] mostraron diferentes estudios para comprender las formas de optimizar el rendimiento de varios sistemas de almacenamiento para *Big Data Warehousing*. Se mencionó que pocos de ellos exploran el impacto de las estrategias de organización de datos en el rendimiento de las consultas cuando utilizan Hive como tecnología de almacenamiento para implementar sistemas de *Big Data Warehousing*. Por esta razón, los autores evaluaron el impacto de la partición y el almacenamiento de datos en sistemas basados en Hive, probando diferentes estrategias de organización de datos y verificando la eficiencia de esas estrategias en el rendimiento de las consultas. Como conclusión se mencionó que la implementación de estrategias basadas en la fragmentación trae beneficios tanto en términos de almacenamiento como en términos del procesamiento de las consultas. Además, se presentaron buenas prácticas que se infieren del análisis realizado, resaltando que se preste especial atención a una excesiva fragmentación, ya que los sistemas presentan un declive en términos de rendimiento.

En los sistemas de bases de datos distribuidas, los costos de comunicación y el tiempo de respuesta han sido desafíos abiertos durante mucho tiempo. Sin embargo, cuando los DDBS (*Distributed Dabase System*, Sistema de Bases de Datos Distribuidas) se diseñan cuidadosamente, se logra la reducción deseada en los costos de comunicación. En [26] se introdujo un enfoque heurístico de *k-means* para la fragmentación vertical y la asignación. Este enfoque se centró en el diseño de DDBS en la etapa inicial. Se llevó a cabo un estudio experimental breve pero efectivo, tanto en conjuntos de datos creados artificialmente como reales, para demostrar la optimización del enfoque propuesto frente a sus homólogos. Los resultados obtenidos sustentaron que el trabajo mostrado por el autor superó a diferentes propuestas en la etapa de experimentación.

Debido a que las empresas utilizan la computación en la nube, esta es esencial para tener los datos y aplicaciones en servidores remotos. Por el rendimiento y eficacia que ofrecen, la fragmentación, asignación y replicación de datos se convierten en algo necesario para llevar a cabo la distribución de los datos. El problema es que los enfoques anteriores que sugieren soluciones de fragmentación vertical se basan en la frecuencia de las consultas de los usuarios, estos enfoques tienen la

limitación de una mayor complejidad y disponibilidad de la frecuencia de las consultas del usuario en la etapa inicial del diseño de la base de datos, aunada a la partición binaria iterativa en el caso de la partición n-aria que aumentara la complejidad y el valor umbral predeterminado que se utiliza para mejorar los resultados, así como el problema de más cálculos. Raouf, Badr y Tolba [27] propusieron un esquema completo de fragmentación, asignación y replicación vertical llamado FVFAR (*full vertical fragmentación, allocation and replication*, Fragmentación vertical, asignación y replicación completa). que realiza el trabajo en un entorno de servicio en la nube, abarca limitaciones de las soluciones de la fragmentación vertical y proporciona asignación y replicación vertical como un servicio en la nube. FVFAR comienza desde la fase de análisis de requisitos del ciclo de vida de desarrollo del sistema para dividir las relaciones de la base de datos distribuida verticalmente en la etapa inicial de diseño de la base de datos distribuida, sin la necesidad de la frecuencia de consultas de los usuarios que no están disponibles en esta etapa, también asigna y replica los fragmentos resultantes a los sitios de la base de datos distribuida mejorando el rendimiento del sistema, aumentando la disponibilidad y reduciendo el costo de comunicación del acceso a la base de datos.

Buscando cómo reducir el costo de transmisión (TC por sus siglas en inglés) en las consultas en sistemas de bases de datos distribuidas, la fragmentación, la agrupación de sitios y la distribución de datos se consideran las principales alternativas para reducción del TC. En [28] los autores presentaron la metodología para fragmentación de datos y asignación ASGOP (A de agregado, S de basado en similitud, GO de orientado a lo voraz y P de aproximado) con la cual obtuvieron resultados favorables en la agrupación y fragmentación de datos, presentaron un algoritmo de agrupación de sitios y dos algoritmos de asignación de datos en sitios, lograron identificar cuál es la tarea de mayor impacto en el diseño de los DDBS, concluyeron que la agrupación de los sitios es responsable de reducir el TC. Para el desarrollo de su trabajo, programaron en C++ en una computadora que tenía un procesador Intel(R) Dual-Core (TM) i3 a 1.7 Ghz con 2 Gb de RAM y 80 Gb de disco duro.

Dahal y Joshi [29] buscaron una estrategia de diseño para sistemas de gestión de bases de datos distribuidas, la principal contribución de este trabajo de investigación incluye la utilización de la técnica de fragmentación vertical basada en agrupamiento donde los atributos de la tabla con similitud residen en un mismo grupo de fragmentos. Este método propuesto utiliza la información relativa a la frecuencia de consulta del usuario para generar una matriz de afinidad de atributos.

En [30], los autores trataron de solucionar los problemas que tiene una tienda de maquinaria agrícola en los tiempos de atención a clientes por parte del personal que pudiera tener problemas con el reconocimiento de artículos. Propusieron un sistema CBIR que permitió reducir los tiempos de atención al cliente, su sistema utiliza el descriptor SURF (*Speeded Up Robust Features*) que obtuvo un mejor resultado que SIFT (*Scale-Invariable Feature Transform*) en eficiencia y efectividad en la recuperación de las imágenes.

Los investigadores de [31] se centraron en reducir el tiempo de respuesta de las consultas basadas en contenido, proporcionando un sistema híbrido de recuperación de imágenes en el que los atributos de textura, color y forma de una imagen se eliminan mediante el uso de la matriz de co-ocurrencia de nivel de gris (GLCM), el momento del color y el procedimiento de accesorios de región, respectivamente. Luego, las características fusionadas extraídas se seleccionaron de manera óptima mediante el análisis de componentes principales (PCA). Posteriormente, probaron dos tipos de técnicas de indexación, a saber, la indexación basada en similitudes y la indexación basada en grupos, en el sistema híbrido desarrollado para encontrar la mejor entre ellas. Los resultados del descriptor de color híbrido basado en la técnica de indexación basada en clústeres mostraron que el sistema propuesto obtuvo mayor precisión.

En [32] se propuso un método heurístico para optimizar la fragmentación vertical en bases de datos distribuidas, utilizaron un algoritmo de agrupamiento y un modelo de costos de transmisión para mejorar el rendimiento de la base de datos y lograron que la técnica reduzca el acceso a fragmentos irrelevantes mediante un algoritmo de asignación de fragmentos de naturaleza voraz, permitiendo reducir el costo de transmisión eficazmente.

Los autores en [33] presentaron un algoritmo para fragmentación vertical llamado, SVP (*Support-Based Vertical Partitioning*) que consiste en tres pasos, el primero tiene como finalidad obtener una matriz de soporte de atributos (ASM) mediante la matriz de uso de atributos (AUM), en el segundo paso, se determina el soporte mínimo automáticamente, y el último paso utiliza un algoritmo de partición basado en conexión para encontrar los fragmentos óptimos. Como resultado, obtienen que su algoritmo en cada experimento encuentra la fragmentación óptima mediante la generación automática del umbral de soporte mínimo.

Rodríguez-Arauz et al. [34] propusieron un método de fragmentación horizontal que optimice consultas basadas en contenido en una base de datos multimedia del Instituto Tecnológico de Orizaba, solucionan la problemática de la gestión de datos históricos de la escuela, utilizaron un modelo de costos que reduce el acceso a tuplas irrelevantes permitiendo una reducción del costo de ejecución de las consultas basadas en contenido.

2.2 Análisis comparativo

En la Tabla 2.1 se muestra un análisis comparativo de los trabajos anteriormente descritos, para que se observen las diferencias y las similitudes entre ellos de una mejor manera.

Tabla 2.1 Tabla comparativa de los trabajos del estado del arte

Artículo	Problema	Contribución	Tecnologías	Resultados
[13]	Minimizar la cantidad de accesos a datos irrelevantes y el costo de transporte de las consultas en bases de datos multimedia distribuidas para lograr una recuperación eficiente de objetos multimedia.	Un algoritmo de partición vertical para bases de datos multimedia distribuidas (MAVP, *Multimedia Adaptable Vertical Partitioning*).	No se menciona.	Una favorable fragmentación mediante el algoritmo propuesto, el algoritmo es eficiente porque toma en cuenta el tamaño de los atributos, propusieron un modelo de costos que considera el costo total de procesamiento de consultas en un entorno multimedia distribuido que consiste en el costo de atributos irrelevantes y el costo de transporte.
[18]	Buscan realizar una fragmentación vertical óptima en bases de datos distribuidas utilizando un algoritmo de energía de enlace diferencial.	Se propuso un nuevo algoritmo de energía de enlace diferencial (DBE, *differential bond energy*) con el objetivo	Python 3, procesador Intel Core i5 de 1,6 GHz, 8 GB de RAM DDR3 de 1600 MHz,	Los resultados mostraron que el algoritmo propuesto es apto para la fragmentación de conjuntos de datos de alta dimensión y encuentra fragmentos de buena calidad en la

Artículo	Problema	Contribución	Tecnologías	Resultados
		de determinar el punto de partición óptimo.	sistema operativo Mac OS High Sierra.	fragmentación vertical del diseño de bases de datos distribuidas.
[19]	Obtener el diseño eficiente de un sistema de bases de datos distribuidas, por medio de la fragmentación y asignación para mejorar su desempeño.	El algoritmo BEA (*Bond Energy Algorithm*).	No se menciona.	El uso del proceso jerárquico dio como resultado la agrupación de conjuntos de atributos más similares y una mejor fragmentación de datos.
[22]	Diseñar un algoritmo de agrupación de atributos apropiado para lograr un rendimiento óptimo de procesamiento de datos en el entorno de Hadoop orientado a columnas	Un nuevo algoritmo llamado CHAC (*Column-oriented Hadoop based Attribute Clustering*)	Hadoop-0.20.2, Mastiff-0.1.2 and Hive-0.5.0	Los resultados generados por el modelo de costos están estrechamente relacionados con el tiempo de ejecución de las consultas en las fases del mapeo, ya que su tendencia es consistente, lo que indica la efectividad del modelo de costos propuesto. CHAC redujo el tiempo de ejecución de las consultas 9.8% en un contexto no traslapado y 19.5% en un contexto traslapado.
[23]	Procesamiento de datos brutos con base en consultas con carga parcial de atributos.	Una formulación lineal de optimización de programación de enteros mixtos que demostró ser *NP-hard*, se diseñó una heurística con dos etapas que encuentra una solución cercana a la óptima en una fracción del tiempo.	No se menciona	Los resultados confirmaron el rendimiento superior de la heurística propuesta sobre los algoritmos de fragmentación vertical relacionados y la precisión de la formulación al capturar los detalles de ejecución de un operador real.
[24]	Encontrar un esquema de fragmentación vertical adecuado para una carga de trabajo.	Se amplió el trabajo *GridFormation*, asignando la tarea de RL (*Reinforcement learning*, Aprendizaje de Refuerzo).	No se menciona.	Se mostró como resultado que se descubrió que el algoritmo de fuerza bruta es superado (en tiempo de optimización) por la solución propuesta, y que esta es competitiva con algoritmos de vanguardia, ya que permanece en el mismo orden de magnitud y se vuelve más competitiva cuando el número de atributos aumenta.
[25]	Optimizar el rendimiento de varios sistemas de almacenamiento para *Big Data Warehousing*	Presentaron buenas prácticas que se infieren del análisis realizado	No se menciona.	Como resultados se obtuvieron recomendaciones basadas en el análisis presentado. Las buenas prácticas mencionadas tuvieron cinco enfoques: general, para estudios posteriores, para la implementación de técnicas de fragmentación, para la

Artículo	Problema	Contribución	Tecnologías	Resultados
				implementación de técnicas de *bucketing* y para el desempeño.
[26]	Reducir los costos de comunicación y tiempo de respuesta en los sistemas de bases de datos distribuidas.	Un enfoque heurístico de *k-means* para la fragmentación vertical.	No se menciona.	Los resultados obtenidos sustentaron que el trabajo mostrado por el autor superó a diferentes trabajos en la etapa de experimentación.
[27]	En la etapa inicial del diseño de la base de datos, no se cuenta con consultas de usuario necesarias para realizar la fragmentación vertical de la base de datos.	Un esquema de fragmentación vertical, asignación y replicación para bases de datos en la nube llamado FVFAR.	No se menciona.	Como resultados obtuvieron una reducción de los costos totales de comunicación para ejecutar las consultas, también que el enfoque propuesto fragmenta verticalmente, asigna y replica los fragmentos resultantes a los sitios de la base de datos distribuida sin que el diseñador tenga que esperar por datos empíricos sobre frecuencias de consultas de la base de datos.
[28]	Reducción de los costos de trasmisión de datos entre los sitios de red de un sistema de base de datos distribuida.	ASGOP que es un esquema de trabajo para fragmentar verticalmente, replicar y asignar fragmentos a sitios.	Lenguaje de programación C++, una computadora equipada con un procesador Intel i3.	Con su método de asignación de datos lograron reducir el costo de transmisión de datos, así como aumentar el rendimiento de la base de datos distribuida.
[29]	Resolver el problema y la complejidad que se produce en el enfoque de fragmentación vertical.	Una técnica de fragmentación basada en agrupamiento en la que los atributos de una tabla con mayor similitud residen en el mismo grupo de fragmentos.	No se menciona.	El método propuesto en este trabajo de investigación benefició el proceso de fragmentación de dos maneras, la primera, en reducir la complejidad que conlleva el aumento del número de sitios, la segunda, se utilizó fragmentación iterativa para evitar la complejidad de la partición iterativa n-aria.
[30]	Mejorar los tiempos de servicio al cliente y reducir la dependencia de un experto.	Sistema que identifica diferentes productos, para la empresa llamada AGROMAQ.	-BoofCV -SURF -SIFT	Con las pruebas concluyeron que SURF es más rápido que SIFT y tiene mejor eficiencia y eficacia.
[31]	Reducir los tiempos de recuperación de imágenes en colecciones de datos enormes.	Un sistema de recuperación de imágenes eficaz con una técnica de indexación	MATLAB R2018a, Procesador core i3, 4 GB de memoria,	Los resultados obtenidos se compararon con muchas técnicas de vanguardia en términos de precisión promedio, también se comparó el tiempo de recuperación con muchas

Artículo	Problema	Contribución	Tecnologías	Resultados
		para reducir el tiempo de recuperación.	Windows de 64 bits.	técnicas relacionadas para evaluar los resultados obtenidos.
[32]	Encontrar una solución más óptima para el problema de fragmentación, replicación y asignación de las bases de datos distribuidas.	Un algoritmo de fragmentación de datos basado en una similitud agregada con el fin de maximizar la coincidencia entre el conjunto de consultas en consideración.	No se menciona.	Se espera que esta técnica pueda reducir considerablemente los costos de transmisión de consultas distribuidas.
[33]	Encontrar un esquema de fragmentación vertical óptimo.	Un algoritmo para fragmentación vertical llamado, SVP (*Support-Based Vertical Partitioning*).	No se menciona.	El algoritmo en cada experimento encontró la fragmentación óptima mediante la generación automática del umbral de soporte mínimo.
[34]	Manejo de la información multimedia de una institución educativa.	Propone la creación de un sistema de gestión de datos multimedia que utilice un método de fragmentación horizontal.	MongoDB, Java Server Faces, NetBeans, BoofCV.	Se demostró que se redujo el costo de ejecución de consultas basadas en contenido en la base de datos multimedia HITO.

Se concluye, después de analizar los anteriores trabajos, que la mayor parte de los artículos se centran en la fragmentación vertical para base de datos relacionales, por otro lado, se presentaron técnicas de recuperación de imágenes con base en contenido. En [34] se desarrolló un sistema para la gestión de datos multimedia que utiliza fragmentación horizontal y considera CBIR. En [13] se propuso un método de fragmentación vertical para base de datos multimedia pero no toma en cuenta las consultas basadas en contenido, del resto de artículos, algunos se enfocan en fragmentar verticalmente base de datos relacionales, NoSQL, o en búsqueda de datos multimedia basada en contenido pero no en un método de fragmentación vertical. Es por esto que en este trabajo se realizó un método de fragmentación vertical para base de datos multimedia que considera búsqueda basada en contenido, además se basa en el costo por atributo que permite asignar a sitios de la red los atributos que hayan sido más utilizados en ese nodo.

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.

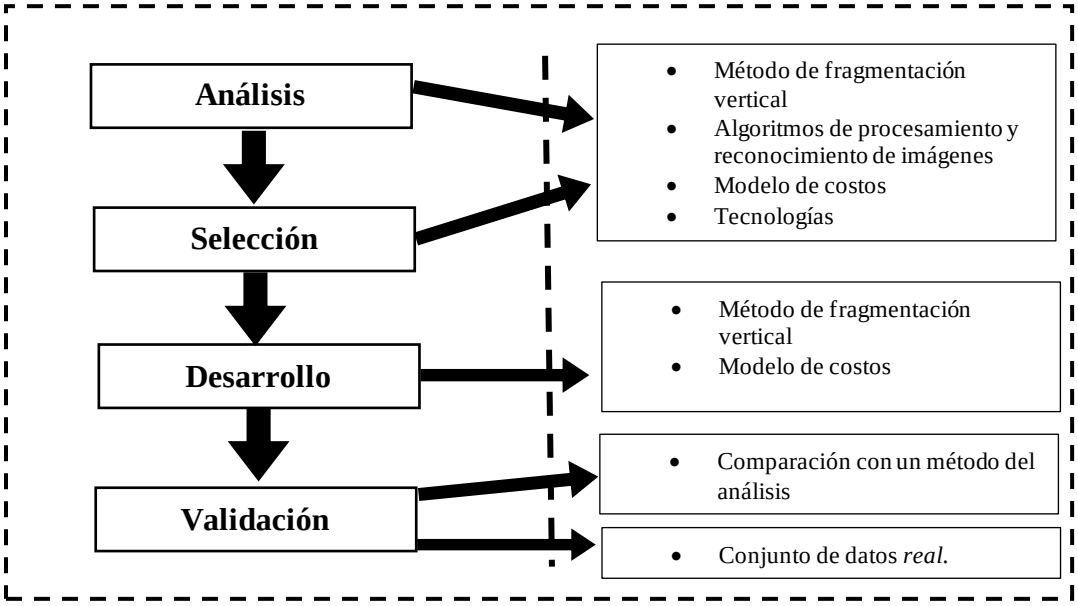


Figura 2.1 Estructura de la propuesta de solución

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

1 **Análisis:** En esta etapa del trabajo se analizaron diferentes artículos científicos por medio de la búsqueda en las principales bibliotecas digitales como ACM Digital Library, SpringerLink, SciencieDirect y IEEE Xplore, donde se encuentren métodos de fragmentación vertical, algoritmos de procesamiento y reconocimiento de imágenes, modelos de costo y tecnologías empleadas para su desarrollo.

2 **Selección:** Durante esta etapa se seleccionó un método de fragmentación vertical para compararlo con el propuesto, así como un algoritmo de reconocimiento y procesamiento de imágenes, un modelo de costos y tecnologías (lenguaje de programación, IDE,

frameworks, sistema gestor de bases de datos, entre otras) de los trabajos anteriormente analizados. El método seleccionado fue el que cumplió con tener una solución completa, que considere la fragmentación vertical, que contenga un modelo de costos, que tenga facilidad de implementación y, por último, que se enfoque en mejorar el desempeño de las consultas.

3 **Desarrollo:** En esta etapa se desarrolló el método de fragmentación vertical para bases de datos multimedia que considere consultas basadas en contenido y el modelo de costos utilizando las tecnologías seleccionadas en la fase anterior.

4 **Validación:** El objetivo de esta etapa es comparar el método de fragmentación y el modelo de costos desarrollados con el trabajo seleccionado del análisis comparativo por medio de un conjunto de datos reales. De esta manera, se comprobó 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	Postgres-XL

Java es una tecnología que se usa para el desarrollo de aplicaciones que convierten a la Web en un elemento más interesante y útil. Java permite jugar, cargar fotografías, chatear en línea, realizar visitas virtuales y utilizar servicios como, por ejemplo, cursos en línea, servicios bancarios en línea y mapas interactivos [35].

JSF (*JavaServer Faces*) es el marco estándar de interfaz de usuario (UI) orientada a componentes para la plataforma Java EE (*Enterprise Edition*), en términos más familiares es un marco web basado en Java.

JSF está incluido en la plataforma Java EE, por lo cual es posible crear aplicaciones que usen JSF sin agregar bibliotecas adicionales en el proyecto. JSF funciona igualmente bien como un marco web independiente capaz de conectarse a contenedores de *beans* como Spring.

Mas allá de un marco web, JSF ofrece un ecosistema de bibliotecas y extensiones de componentes de UI portátiles. Esto significa que se puede tomar un componente de la interfaz de usuario de una biblioteca, como árbol o una tabla desplazable y usarlo en cualquier implementación de JSF. La mayoría de las bibliotecas de componentes de la interfaz de usuario se centran en proporcionar componentes de interfaz de usuarios enriquecidos, lo cual implica Ajax.

JSF tiene dos funciones principales. La primera es generar una interfaz de usuario, normalmente una respuesta HTML que se envía a un navegador y se ve como una página web. Esta interfaz es representada en el servidor por un árbol de componentes. La interfaz de usuario real se genera cuando el árbol de componentes está codificado (o renderizado). Esta separación entre el árbol de componentes y la interfaz de usuario permite que JSF admita diferentes lenguajes de marcado (HTML vs XUL) o entornos de navegador alternativos (escritorio o teléfonos inteligentes).

La segunda función de JSF es responder a eventos generados por el usuario en la página invocando los oyentes del lado del servidor, seguido de la generación de otra interfaz de usuario o una actualización de la interfaz de usuario ya mostrada (posiblemente a través de Ajax). En este sentido, se dice que JSF es un marco web impulsado por eventos.

Es importante mencionar que JSF está integrado en cualquier servidor de aplicaciones Java EE compatible, como *WebLogic* de Oracle, *GlassFish Open Source Edition* o JBoss As, pero también se puede utilizar como una biblioteca independiente de contenedores de *servlets* como *Tomcat* y *Jetty* [36].

La plataforma de NetBeans es básicamente un *framework* que simplifica el desarrollo de aplicaciones de escritorio Java. Es capaz de instalar módulos de forma dinámica. Además de Java, también admite otros lenguajes, incluido PHP, C, C++ y HTML 5.

NetBeans es un entorno de desarrollo integrado de código abierto. Proporciona modularidad al código, ya que admite un enfoque modular, es decir, permite que las aplicaciones se desarrollen como módulos (como componente de un software).

Con su editor Java que mejora constantemente, muchas funciones completas y una amplia gama de herramientas, plantillas y muestras, NetBeans IDE establece el estándar para el desarrollo con tecnologías de vanguardia listas para usar [37].

UWE (*Unified Modeling Language Web Engineering*, Ingeniería Web del Lenguaje Unificado de Modelado) es una metodología que permite especificar de mejor manera una aplicación Web en su proceso de creación, mantiene una notación estándar basada en el uso de UML (*Unified Modeling Language*) para sus modelos y sus métodos. La metodología define claramente la construcción de cada uno de los elementos del modelo.

En su implementación se contemplan las siguientes etapas y modelos:

- Modelo de Requerimientos que captura los requerimientos del sistema. Plasma los requisitos funcionales de la aplicación Web mediante un modelo de casos de uso.
- Modelo Conceptual para el contenido. Define, mediante un diagrama de clases, los conceptos a detalle involucrados en la aplicación.
- Modelo de navegación. Representa la navegación de los objetos dentro de la aplicación y un conjunto de estructuras como son índices, menús y consultas.

- Modelo de presentación. Representa las interfaces de usuario por medio de vistas abstractas.
- 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 permiten describir una aplicación Web desde varios puntos de vista abstractos. Cada uno de estos modelos se representa como paquetes UML, dichos paquetes son procesos relacionados que pueden ser refinados en iteraciones sucesivas durante el desarrollo del UWE [38].

Postgres-XL es un clúster de base de datos *SQL* de código abierto escalable horizontalmente, lo suficientemente flexible para manejar cargas de trabajo de bases de datos variables.

Los componentes de PostgreSQL son los siguientes:

- **Monitor de transacciones globales (GTM)**

Global Transaction Monitor garantiza la coherencia de las transacciones en todo el clúster. GTM es responsable de emitir los ID de transacción y las instantáneas como parte de su control de concurrencia de múltiples versiones.

El clúster también puede configurarse opcionalmente con un GTM *Standby*, para mejorar la disponibilidad.

Además, es posible configurar un Proxy GTM en los Coordinadores para mejorar la escalabilidad y reducir la cantidad de comunicación con GTM.

- **Coordinador**

El Coordinador gestiona las sesiones de los usuarios e interactúa con GTM y los nodos de datos. El coordinador analiza y planifica las consultas y envía un plan global serializado a cada uno de los componentes involucrados en una declaración.

- **Nodo de datos**

El nodo de datos es donde se almacenan los datos reales. El DBA puede configurar la distribución de los datos. Para mejorar la disponibilidad, se pueden configurar reservas en caliente de los nodos de datos para que estén listos para la conmutación por error.

Postgres-XL (*eXtensible Lattice*) permite fragmentar tablas en múltiples nodos o replicarlas. Las tablas de partición (o distribución) permiten la escalabilidad de escritura en varios nodos, así como el procesamiento masivo en paralelo (MPP) para cargas de trabajo de tipo Big Data.

Las tablas replicadas suelen ser datos estáticos que no cambian con mucha frecuencia. Replicarlos permite la escalabilidad de lectura.

Postgres-XL es una base de datos transaccional totalmente compatible con ACID (*Atomicity, Consistency, Isolation, Durability*, atomicidad, consistencia, aislamiento, durabilidad) que no solo proporciona una vista totalmente coherente de los datos en todo momento, sino que también utiliza el MVCC (*Multi-Version Concurrency Control*, Control de concurrencia de múltiples versiones) en todo el clúster. Cuando inicie una transacción o consulta en Postgres-XL, se verá una versión coherente de los datos en todo el clúster. Al leer los datos en una conexión, es posible actualizar la misma tabla o incluso una fila en otra conexión sin ningún bloqueo. Ambas conexiones funcionan con sus propias versiones de las filas, gracias a los identificadores de transacciones globales y las instantáneas. Los lectores no bloquean a los escritores y los escritores no bloquean a los lectores [39].

Capítulo 3. Aplicación de la metodología

Este capítulo aborda la metodología seguida para la realización del método de fragmentación vertical para bases de datos multimedia.

3.1 Análisis

Conforme a uno de los objetivos específicos, se realizó un análisis profundo de los trabajos relacionados con la fragmentación vertical de bases de datos multimedia. Se llevó a cabo una búsqueda dentro de este análisis en algunas de las principales bibliotecas digitales de editoriales: ACM, IEEE, Springer y Elsevier.

Para realizar el estudio de los trabajos relacionados se siguió la metodología descrita a continuación en la Figura 3.1.

En la Figura 3.1 se muestran todas las etapas de la metodología propuesta. Como ya se mencionó, se realizó la búsqueda de trabajos en las principales bibliotecas digitales de editoriales científicas, ACM, IEEE, Springer y Elsevier. Los trabajos encontrados que no son publicados por dichas editoriales se categorizan en "Otras". La búsqueda consiste en encontrar todos los trabajos que contengan las siguientes palabras clave: *vertical fragmentation (fragmentación vertical)*, *cost model* (modelo de costo) y CBIR. Los trabajos deben haberse publicado entre los años 2010 y 2020. Ya obtenidos todos los trabajos, se aplicó un filtro y se descartaron todos los trabajos que sean tesis de maestría o doctorado, así como libros y artículos que no estén escritos en inglés. Los artículos resultantes se clasificaron por editorial y por año. En esta última etapa se analizó cada uno de ellos bajo seis rubros principales: Fragmentación vertical, Completitud, Facilidad de implementación, Modelo de costos, Consultas basadas en contenido y Tipo de base de datos. De esta manera, sólo los trabajos que cumplieron con estos seis rubros se seleccionaron.

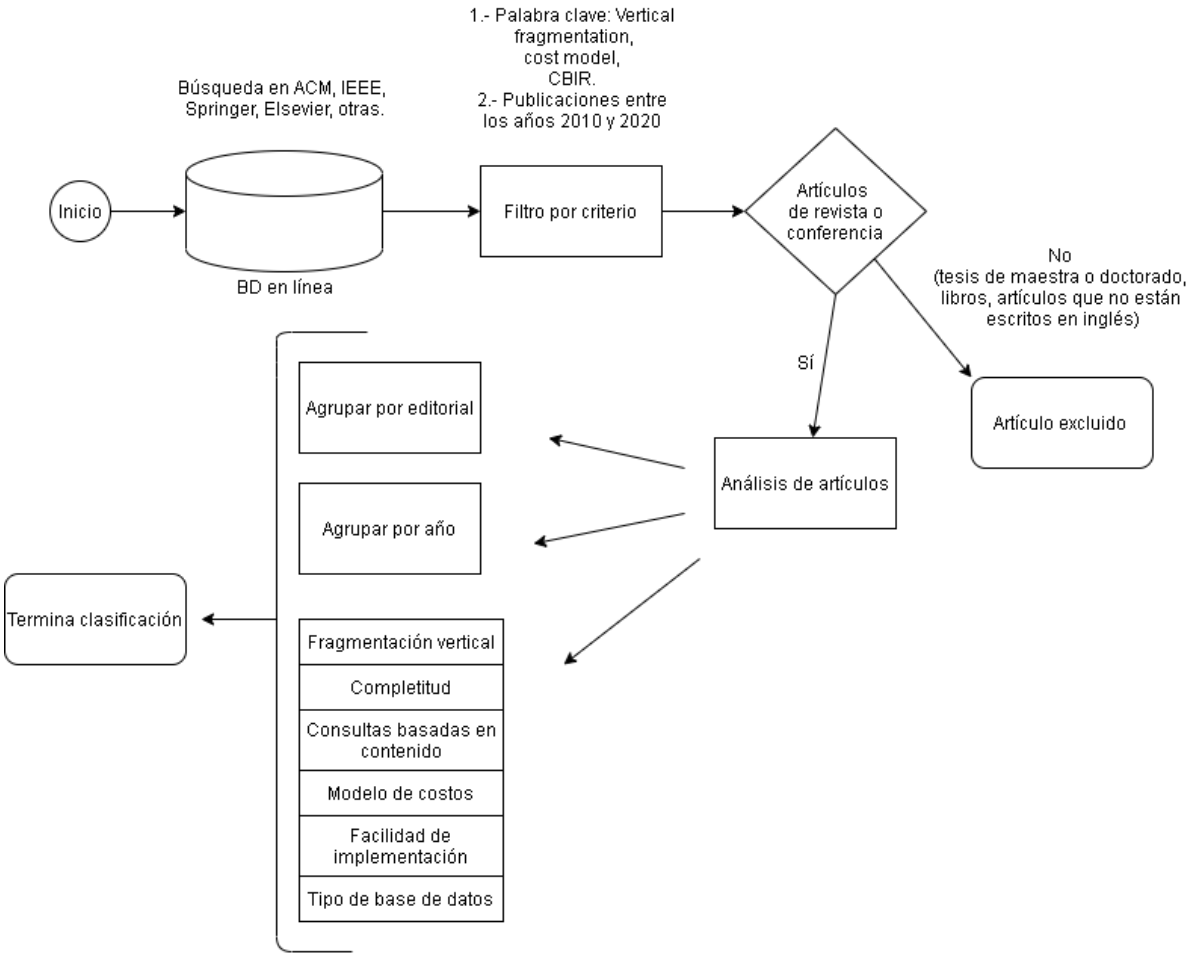


Figura 3.1. Metodología de búsqueda y evaluación de los trabajos relacionados

Se observa en las Tablas 3.1 a 3.5 la comparación de los trabajos encontrados en las bibliotecas científicas digitales. Cada artículo se evaluó utilizando una metodología para determinar si cada artículo cumple con cada una de las características.

Tabla 3.1. Comparación de los trabajos relacionados de la editorial ACM

1	2	3	4	5	6	7
Zhao et al. [23]	X	X	X			Big data
Abdel Raouf et al. [27]	X	X		X		Distribuida
Amer et al. [40]	X			X		Distribuida
1) Artículo, 2) Fragmentación vertical, 3) Completitud, 4) Fácil de implementar, 5) Modelo de costo, 6) Consultas basadas en contenido, 7) Tipo de base de datos.						

Tabla 3.2. Comparación de los trabajos relacionados de la editorial IEEE

1	2	3	4	5	6	7
Gu et al. [22]	X		X	X		Big data
Dahal y Joshi [29]	X					Distribuida
Amer et al. [32]	X	X		X		Relacional
Rodriguez y Li [33]	X	X	X			Relacional
Zhang y Zhao [41]	X		X	X		Distribuida
Amer y Abdalla [42]	X	X		X		Distribuida
Chen et al. [43]	X			X		NoSQL
Abdel Raouf et al. [44]	X	X				Distribuida
Ho et al. [45]	X	X		X		NoSQL
Kim et al. [46]		X			X	Multimedia
Birhanu et al. [47]	X	X	X	X		Documento XML
Jagannatha et al. [48]	X			X		Distribuida
Amoseen [49]	X	X		X		OLTP
1) Artículo, 2) Fragmentación vertical, 3) Completitud, 4) Fácil de implementar, 5) Modelo de costo, 6) Consultas basadas en contenido, 7) Tipo de base de datos.						

Tabla 3.3. Comparación de los trabajos relacionados de la editorial Elsevier

1	2	3	4	5	6	7
Mehta et al. [18]	X		X			Distribuida
Rahimi et al. [19]	X		X	X		Distribuida
Amer et al. [28]	X	X		X		Distribuida
Huang y Lai [50]	X			X		Distribuida
Pazos et al. [51]	X	X	X	X		Distribuida
1) Artículo, 2) Fragmentación vertical, 3) Completitud, 4) Fácil de implementar, 5) Modelo de costo, 6) Consultas basadas en contenido, 7) Tipo de base de datos.						

Tabla 3.4. Comparación de los trabajos relacionados de la editorial Springer

1	2	3	4	5	6	7
Rodriguez y Li [13]	X	X	X	X		Multimedia
Campero Durand et al. [24]	X	X		X		Relacional
Costa et al. [25]	X	X				Big data
Amer [26]	X			X		Distribuida
Tsuchida et al. [52]	X					Multimedia
Bobrov et al. [53]	X	X	X			Relacional
Kaur y Laxmi [54]	X	X				Relacional
Goli y Rouhani Rankoohi [55]	X			X		Distribuida
Dharavath et al. [56]	X	X				Distribuida
1) Artículo, 2) Fragmentación vertical, 3) Completitud, 4) Fácil de implementar, 5) Modelo de costo, 6) Consultas basadas en contenido, 7) Tipo de base de datos.						

Tabla 3.5. Comparación de los trabajos relacionados de otras editoriales

1	2	3	4	5	6	7
Rojas Ruiz et al. [30]		X	X		X	Multimedia
Bhardwaj et al. [31]		X			X	Multimedia
Rodríguez-Arauz et al. [34]		X	X	X	X	Multimedia
Ghorbanian et al. [57]					X	Multimedia
Kishore y Rao [58]		X			X	Multimedia
Buvana et al. [59]		X			X	Multimedia
Abdalla y Artoli [60]	X	X		X		Distribuida
1) Artículo, 2) Fragmentación vertical, 3) Completitud, 4) Fácil de implementar, 5) Modelo de costo, 6) Consultas basadas en contenido, 7) Tipo de base de datos.						

De todos los trabajos incluidos, se observa que pocos artículos cumplen con las características deseadas. Los trabajos [13] y [34] cumplen con la mayoría de las cualidades, el trabajo [13] no cumple con la característica de considerar consultas basadas en contenido, mientras que, el artículo [34] está enfocado a la fragmentación horizontal.

3.2. Selección

El artículo [13] es fácil de implementar, contiene información completa para llevar a cabo su implementación, contiene un modelo de costos, se puede aplicar a bases de datos multimedia y está enfocado en fragmentación vertical, por ser el método que cumplió con casi todas las características, se decidió compararlo con el método a desarrollar.

La Figura 3.2 muestra el flujo de trabajo del método de fragmentación vertical para base de datos multimedia que toma en cuenta consultas basadas en contenido. En el primer paso, se analiza el archivo de registro de la base de datos, esperando encontrar consultas basadas en contenido. Si se encuentran, la tabla se fragmenta en el segundo paso separando la clave, el descriptor y el atributo multimedia. Al llegar al tercer paso, se analiza la tabla para determinar si tiene columnas que no involucren descriptores (atributos alfanuméricos). Hay dos opciones: 1) Si no existen, en el cuarto paso se verifica si hay más columnas que involucren descriptores (atributos con datos multimedia), si los hay, el flujo vuelve al segundo paso, y si ya no, entonces continúa con el quinto paso donde se crea el esquema con la clave, el atributo y el descriptor. 2) Si existen, en el sexto paso se crea la tabla de costos de atributos. Posteriormente, el séptimo paso consiste en crear el esquema asignando atributos a los sitios donde son más requeridos. Luego, en el octavo paso, los fragmentos se asignan a los sitios y finaliza el flujo. Si la tabla no recibió consultas basadas en contenido, el flujo va directamente al sexto paso.

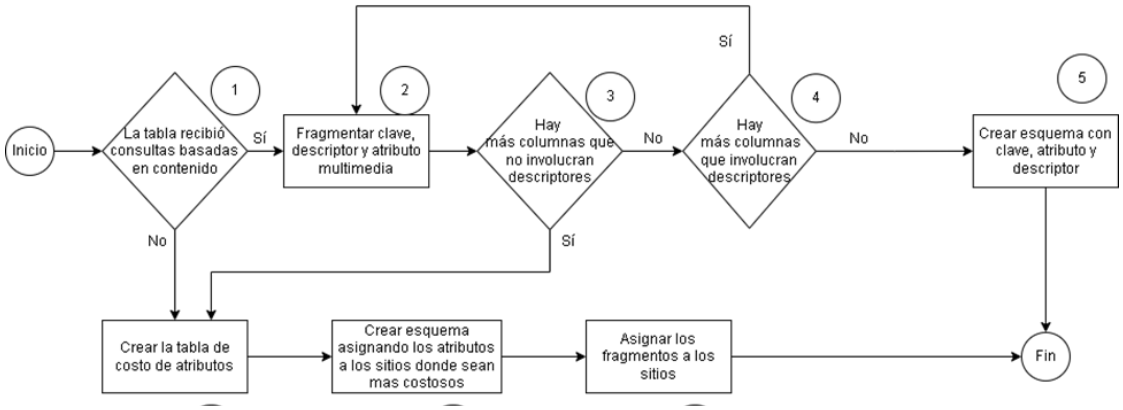


Figura 3.2 Flujo de trabajo del método de fragmentación vertical para base de datos multimedia que toma en cuenta consultas basadas en contenido

3.3 Desarrollo

En esta sección se describe el desarrollo de la aplicación siguiendo la metodología de desarrollo. Se realizan todos los diagramas y pasos de la metodología seleccionada para el desarrollo de la aplicación.

En la Figura 3.3 se representa la arquitectura elegida la cual es "modelo vista controlador" (MVC): en la vista se localizan las páginas que permiten la comunicación del usuario con la aplicación para solicitar información sobre la base de datos y cómo acceder a ella. Los *beans* administrados de JSF (JavaServer Faces) son los encargados de gestionar el flujo de información y se encuentran en el controlador. En la parte de modelo se definen las reglas de negocio, las cuales son encargadas del cálculo del modelo de costos y la detección de características de archivos multimedia para la realización de la fragmentación vertical. La validación de la técnica diseñada se verifica mediante la aplicación web construida para fragmentar verticalmente y asignar fragmentos a sitios más cercanos.

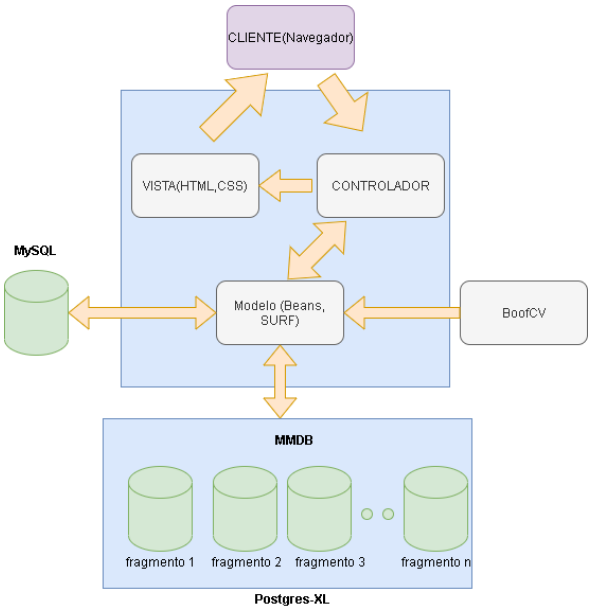


Figura 3.3 Arquitectura de la aplicación Web

3.3.1 Análisis de requisitos

Con la finalidad de establecer los requisitos funcionales de la aplicación web se realizó un análisis de requisitos los cuales se representan mediante diagramas. El diagrama de casos de uso representa la relación que existe entre el actor (administrador de base de datos) y la actividad (Fragmentar y asignar).

En la Tabla 3.6 se define el actor y la descripción del papel que desempeña.

Tabla 3.6 Actor de la aplicación

Actor	Descripción
Administrador de la base de datos (DBA)	El DBA es el único actor en el sistema y su única función es solicitar la fragmentación proporcionando la información necesaria para este proceso

La Figura 3.4 muestra el caso de uso "Fragmentar y asignar" con el estereotipo "*Process*" (Proceso), como lo sugiere la metodología UWE. Este caso de uso se describe con un diagrama de actividades.

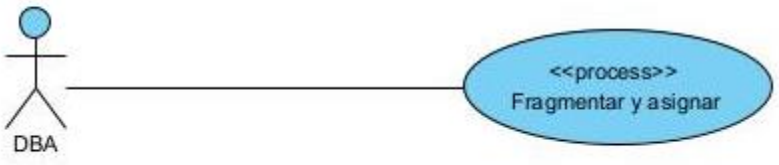


Figura 3.4 Diagrama de casos de uso

En la Figura 3.5 se aprecia el diagrama de actividad del caso de uso que se observa en la Figura 3.4, esta muestra como cuadro la acción del formulario para capturar la información necesaria para llevar a cabo la fragmentación, entre los datos solicitados, se encuentra la dirección de la base de datos, el puerto de acceso, el usuario de la base de datos, su contraseña, el nombre de la base de datos y el nombre de la tabla a fragmentar. El segundo cuadro de acción muestra la actividad del usuario al capturar la información solicitada. La primera decisión es cuando se determina si la tabla recibió consultas basadas en contenido, hay dos casos posibles: el primer caso es cuando la tabla sí recibió consultas basadas en contenido y el sistema toma la acción definiendo los fragmentos que contengan clave, atributo multimedia y descriptor; en la siguiente decisión se verifica si la tabla contiene atributos que no sean multimedia ni descriptores, si existen, entonces el sistema crea la tabla de costos de atributos, pasando al siguiente cuadro, con la información de la tabla de costos, se crea el esquema asignando los atributos a los sitios donde fueron más solicitados, así teniendo el esquema continúa al último cuadro donde se asignan los fragmentos a los sitios llegando al final del diagrama. En el caso de no haber atributos que no involucren descriptores, continúa a verificar sí existen más columnas que involucren descriptores, en el caso de haber más columnas, se regresa a la parte donde se fragmenta por clave, descriptor y atributo multimedia. Si no existieran más columnas que involucraran descriptores, se crea el esquema con clave, atributo multimedia y atributo descriptor para terminar con el flujo. En el caso que la tabla no haya recibido consultas basadas en contenido, se crea la tabla de costos y el esquema para posteriormente asignar los atributos a sitios donde fueron más costos y se colocan los fragmentos en los sitios.

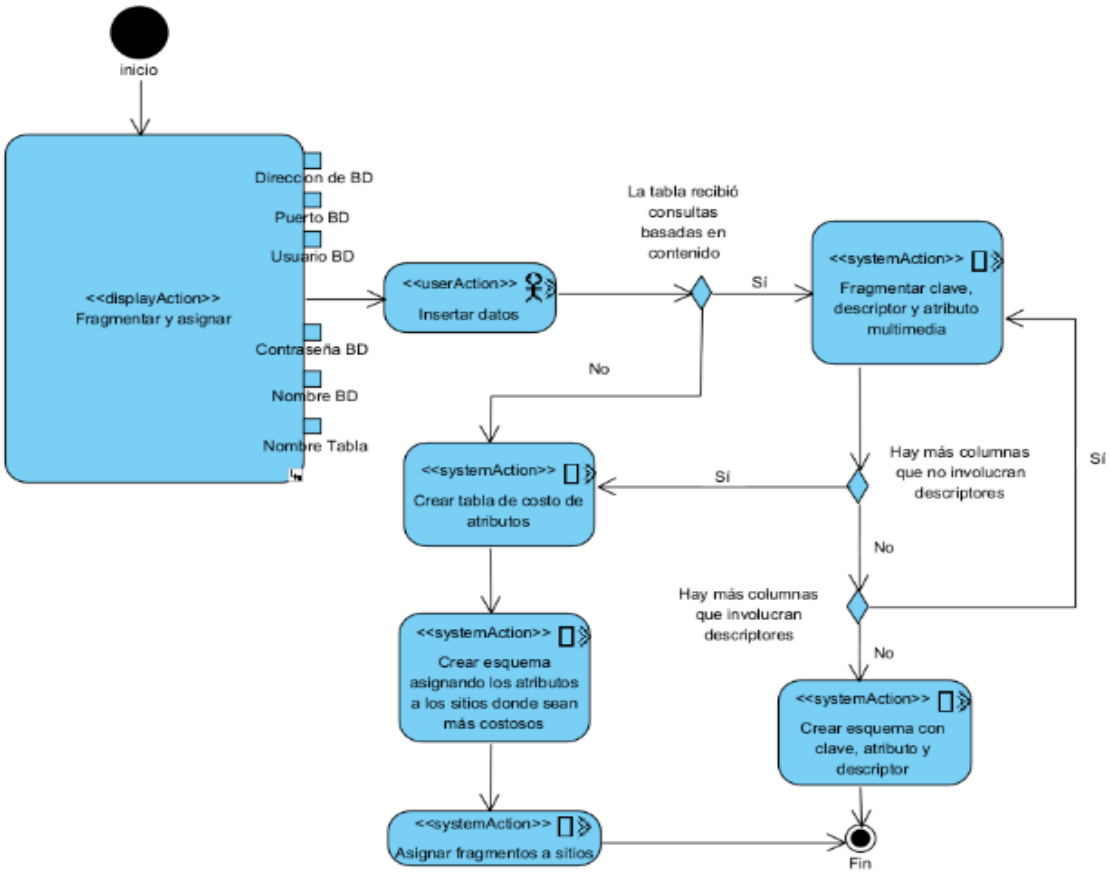


Figura 3.5 Diagrama de actividad

3.3.2 Diseño

A continuación, siguiendo la metodología de desarrollo se presentan los modelos establecidos por la metodología UWE: conceptual, de navegación, de presentación y de procesos.

3.3.2.1 Modelo conceptual

El modelo conceptual, también conocido como modelo de dominio, se encarga de describir cómo se relacionan los requisitos de la aplicación. El modelo conceptual en este trabajo está representado por tres diagramas: diagrama conceptual, diagrama lógico y diagrama físico de la base de datos. Se observan los diagramas en las Figuras 3.6, 3.7 y 3.8.

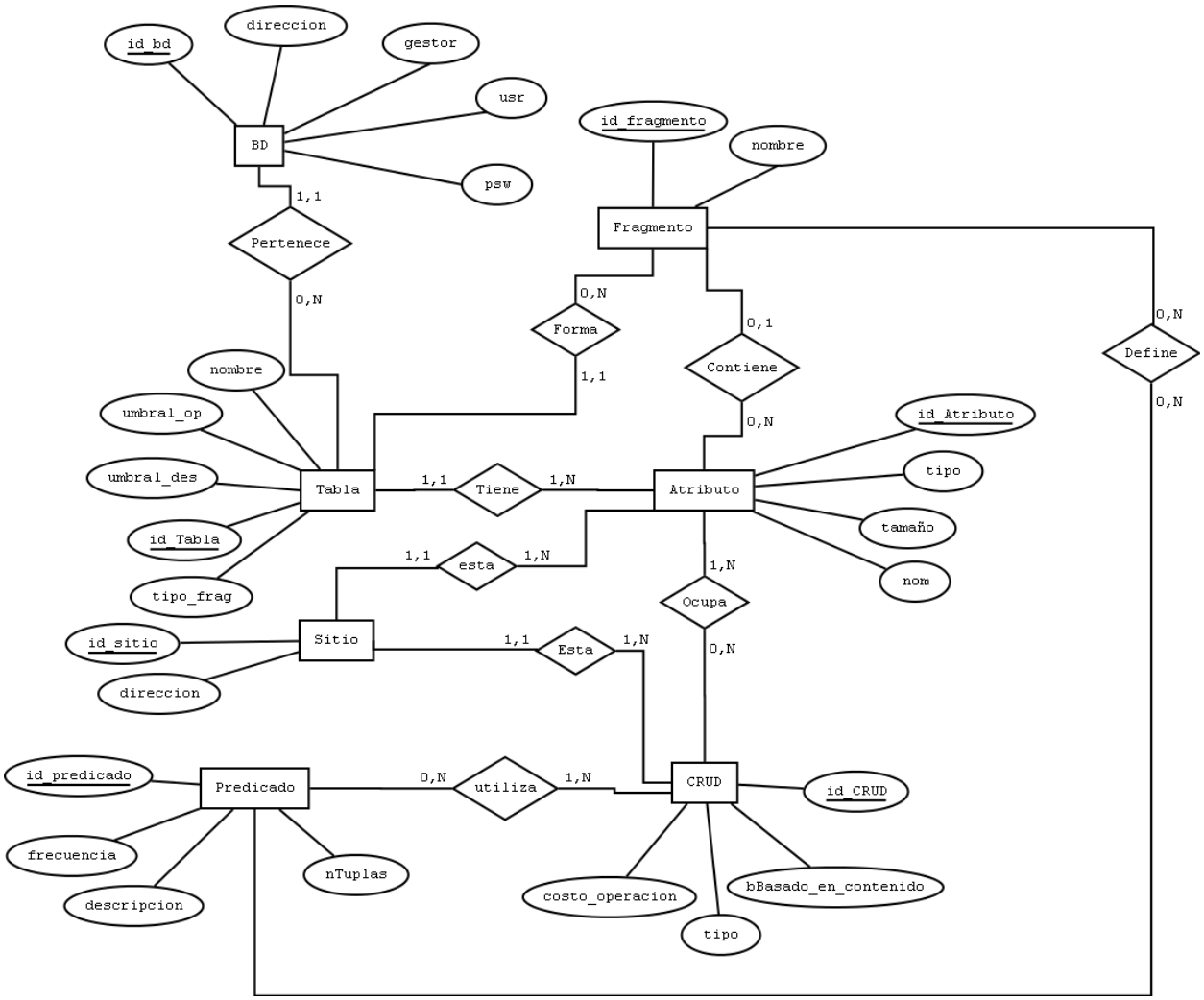


Figura 3.6 Diagrama conceptual de la aplicación

Figura 3.7 Diagrama lógico de la aplicación

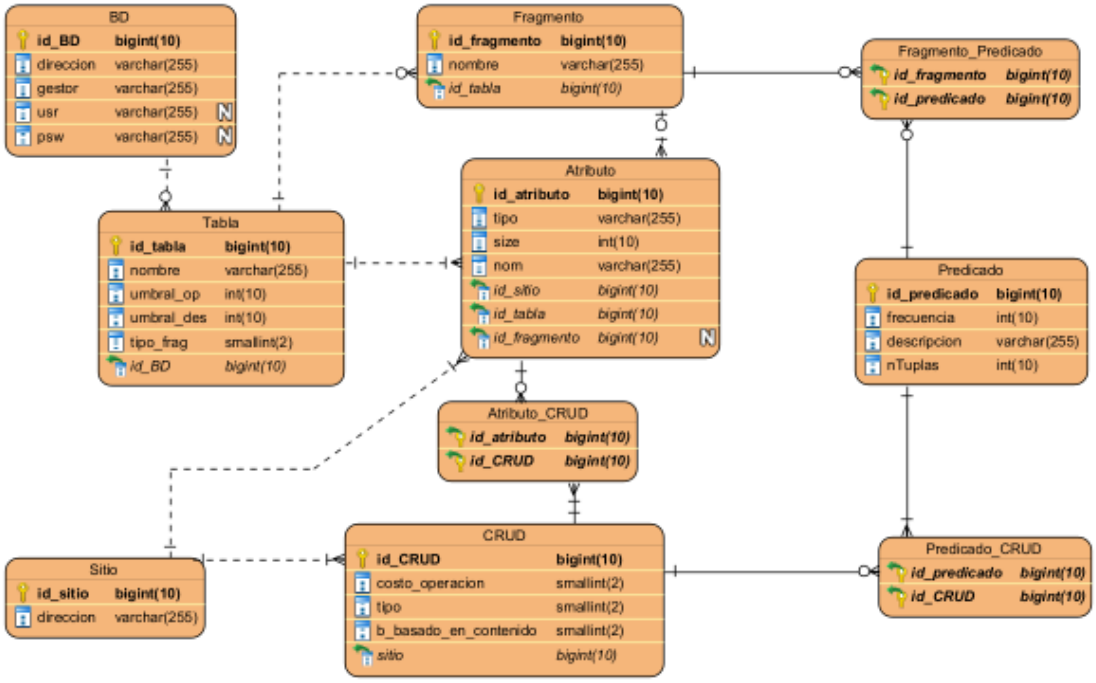


Figura 3.8 Diagrama físico de la base de datos

Tablas de la base de datos

Tabla: La relación *Tabla* es la encargada de guardar la información de las tablas que se ingresan en la información proporcionada por el usuario.

Atributo: La tabla *Atributo* almacena para cada tabla todos sus atributos, indicando su tipo de dato en la columna *tipo*, nombre en la columna *nom* y su tamaño en la columna *size*.

CRUD: La tabla *CRUD* es la encargada de almacenar cualquier operación registrada en el archivo de carga que se relacione con la tabla elegida para realizar la fragmentación. En esta tabla se registra las operaciones de inserción, selección, eliminación o modificación realizadas en la tabla propuesta.

BD: La tabla BD guarda la información general del proceso de fragmentación, como la dirección IP, gestor, usuario y contraseña.

Sitio: En la tabla *Sitio* se guarda la IP del nodo de la red.

Fragmento: La tabla *Fragmento* guarda el nombre del fragmento resultante.

Predicado: La tabla *Predicado* guarda todos los predicados de cada CRUD que se relacionan con la tabla seleccionada por el administrador para ser fragmentada.

Fragmento_Predicado: La tabla *Fragmento_Predicado* guarda la relación entre la tabla *Fragmento* y la tabla *Predicado*.

Atributo_CRUD: La tabla *Atributo_CRUD* relaciona todos los atributos utilizados en cada operación CRUD, de manera que un atributo tiene la oportunidad de estar presente en muchas operaciones CRUD y una operación CRUD puede utilizar muchos atributos.

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 presente en muchos CRUD.

3.3.2.2 Modelo de navegación

Gracias al modelo de navegación se puede conocer los caminos posibles en el recorrido de los usuarios dentro de la aplicación. Este modelo propone realizar mapas para tener claro el recorrido mediante diagramas de clase estereotipados, estos diagramas describen las rutas por las que el usuario puede moverse dentro de una aplicación. En la Figura 3.9 se observa el diagrama de navegación de la aplicación.

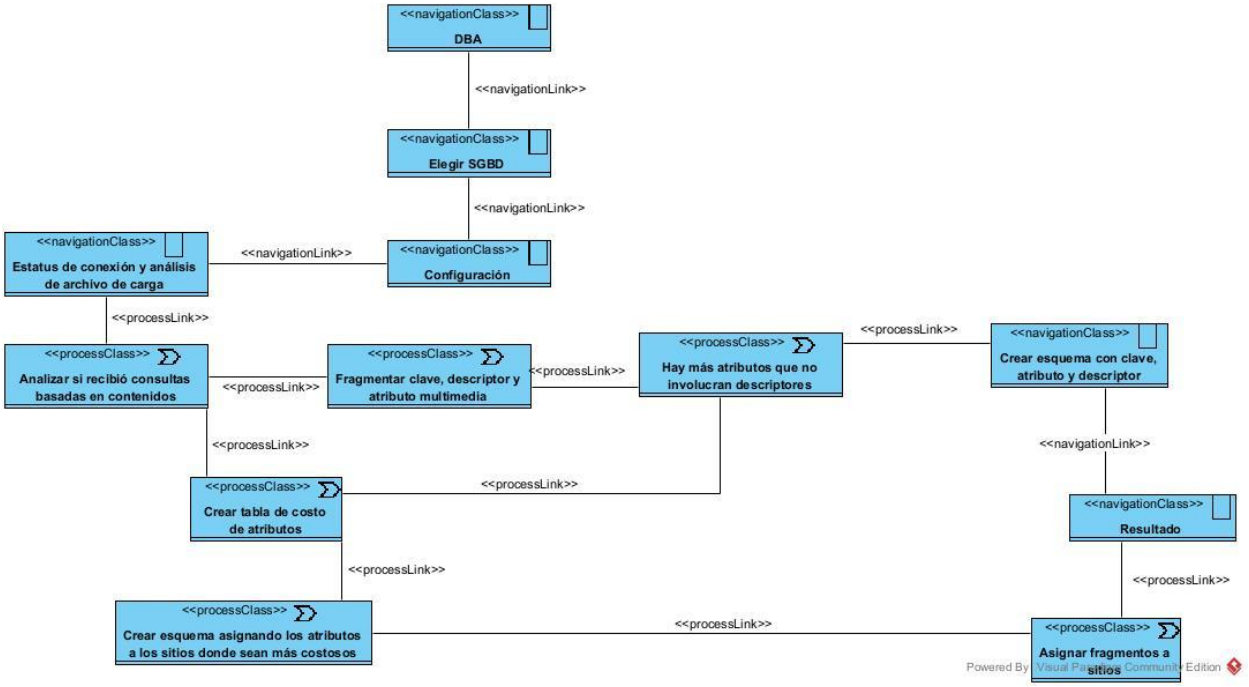


Figura 3.9 Diagrama de navegación

En el diagrama de la Figura 3.9 se describe el modelo navegacional del usuario administrador de la base de datos. Después del rol DBA continúa "*Elegir SGBD*", es la encargada de permitir la elección del tipo de gestor que se conectará en la siguiente clase llamada "*Configuración*", la cual permite al usuario ingresar los datos necesarios 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, dar la información necesaria en caso de considerar consultas basadas en contenido, en caso de no

haber recibido consultas basadas en contenido, se sigue a la clase donde se crea la tabla de costo de atributos, en la siguiente clase se utiliza esta información para asignar los atributos a sitios donde fueron más requeridos y en la siguiente clase se asignan los fragmentos a los sitios llegando a la clase de resultado.

3.3.2.3 Modelo de presentación

Continuando con el diseño y el modelado, en la Figura 3.10 se muestra un modelo de presentación relacionado con la conexión a la base de datos, donde el usuario elige el gestor de bases de datos que aloja la base de datos que se requiere fragmentar, también se observa que después de la elección del gestor de bases de datos, es necesario ingresar la dirección IP, el puerto de conexión, nombre de la base de datos, usuario de la base de datos y su contraseña, también se elige la tabla que se fragmentará.

En el modelo de presentación que se muestra en la Figura 3.11 se aprecia el apartado donde se configura la fragmentación eligiendo la manera de fragmentar, los atributos multimedia si existieran y se puede cargar el archivo de registro de la base de datos, llenando el formulario correctamente, solo queda continuar al presionar el botón *Fragmentar y asignar* para fragmentar la tabla elegida y en el siguiente modelo se presenta el resultado del análisis.

En la Figura 3.12 se observa un modelo de presentación del resultado del análisis y el esquema de fragmentación de base de datos, el usuario administrador de la base de datos puede visualizar la tabla de costo y el esquema realizado, al dar en el botón *Fragmentar y asignar* se comienza a fragmentar, al terminar se muestra un mensaje de "*Fragmentación realizada*".

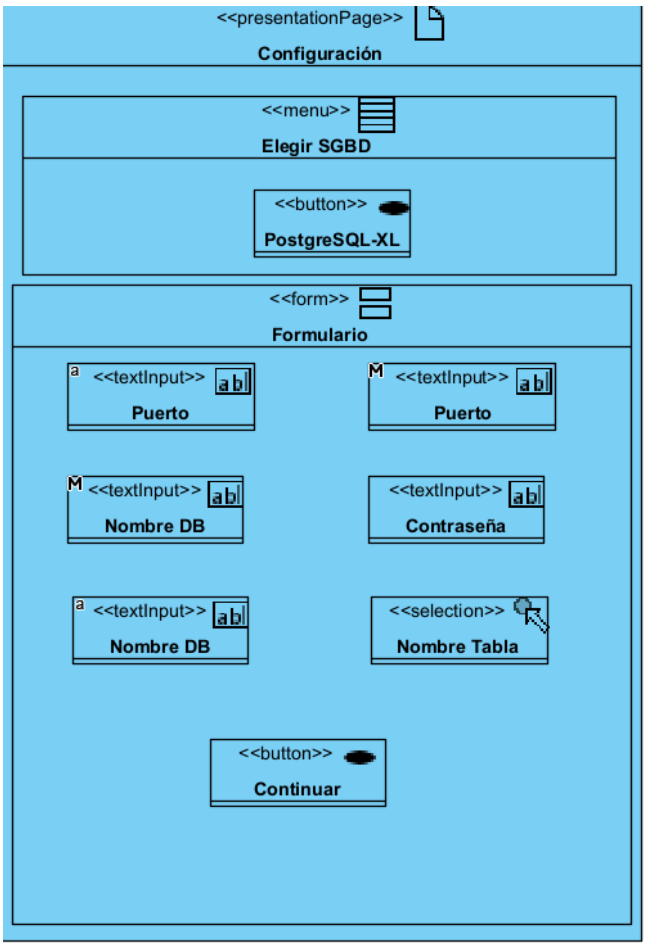


Figura 3.10 Modelo de presentación formulario inicial

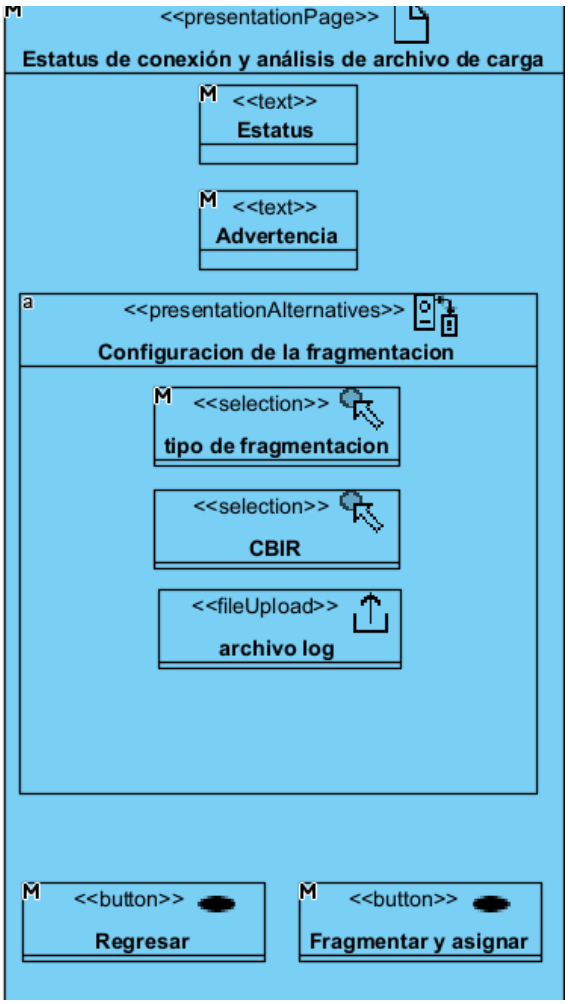


Figura 3.11 Configuración de fragmentación

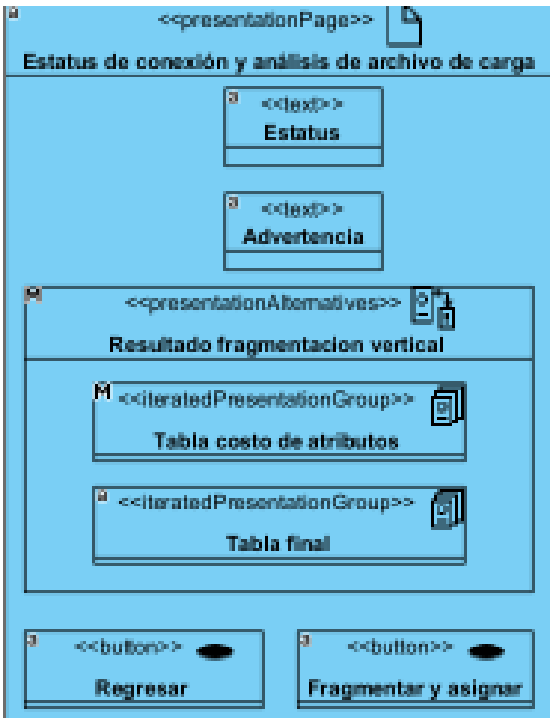


Figura 3.12 Presentación preliminar de la fragmentación

3.3.2.4 Modelo de procesos

El diagrama de proceso que se observa en la Figura 3.13, comienza con el formulario para elegir el tipo de gestor de bases de datos a la que se conectará, su dirección, el usuario dueño de esa base de datos, la contraseña del usuario, el puerto de conexión, el nombre de la base de datos y la tabla que se quiere fragmentar, en caso de que exista conexión, el siguiente paso es configurar la fragmentación, el usuario ingresa el tipo de fragmentación que realizará, el archivo de registro y en caso de que el usuario requiera contemplar consultas basadas en contenido, se elige los atributos de la tabla que son atributos multimedia y atributos descriptores. En el siguiente paso se verifica si recibió consultas basadas en contenido, sí las recibió, se fragmenta por clave, atributo multimedia y atributo descriptor, sí en la tabla existieran columnas que no involucren descriptores, continúa al paso donde se crea la tabla de costos de atributos, con la información de esta tabla se crea un esquema donde los atributos se asignan a sitios donde fueron más requeridos, cuando se

tiene el esquema de fragmentación, se asignan los fragmentos a los sitios y se termina ese proceso. En el caso cuando solo existen atributos que involucren descriptores, se crea el esquema con fragmentos donde están colocados clave, atributo multimedia y atributo descriptor, con este esquema se realiza la fragmentación y se termina el proceso.

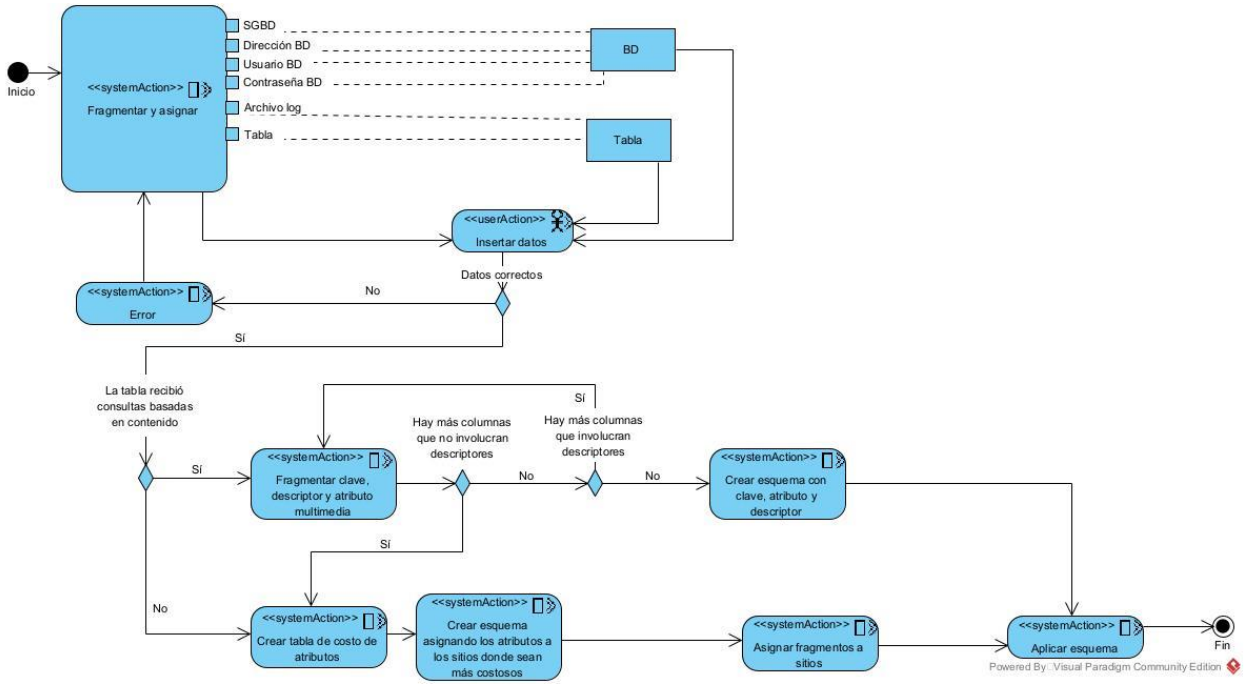


Figura 3.13 Diagrama de fragmentar y asignar del modelo de proceso

3.3.3 Implementación

Una vez finalizado el diseño, se inició con la implementación del método. En la Figura 3.14 se muestra un fragmento de código para la lectura de los archivos de registro de la base de datos, lo que permite tener un arreglo de caracteres y en cada posición existe una línea, para realizar el posterior análisis de consultas.

```
public ArrayList<String> leerTxt(UploadedFiles cad) throws FileNotFoundException, IOException {
        ArrayList<String> lineaArchivo = new ArrayList();

        cad.getFiles().stream().filter((f) -> (f != null && !"".equals(f.getFileName()))).forEachOrdered((f)
-> {
            try (BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(f.getInputStream(),
"UTF-8"))) {
                String actual = "";

                while ((actual = bufferedReader.readLine()) != null) {
                    lineaArchivo.add(actual);
                }

            } catch (Exception e) {
            }
        });

        return lineaArchivo;
    }
```

Figura 3.14 Código de lectura para archivos de registro

Para definir los fragmentos multimedia se utiliza un arreglo de dos posiciones, una posición pertenece al atributo multimedia y la otra al atributo descriptor, esto permite que cada par se asigne a un solo fragmento. La Figura 3.15 contiene el código del método que separa el arreglo en pares para obtener los fragmentos multimedia.

```java
public ArrayList<TablaCosto> obtenerFragMul() {
        ArrayList<TablaCosto> arr = new ArrayList<>();
        ArrayList<atributo> listaAtr;

        for (String str : this.arrAM) {
            listaAtr = new ArrayList<>();
            TablaCosto tabla = new TablaCosto();
            tabla.setSitio(new sitio("localhost", this.direccion));
            listaAtr.add(obteneratributoid());
            for (atributo atr : this.arrAtri) {

                if (str.contains(atr.getNombre())) {
                    listaAtr.add(atr);
                }
            }
            tabla.setArrAtri(listaAtr);
            tabla.setCosto(0);
            arr.add(tabla);
        }

        return arr;
    }
```

Figura 3.15 Separación de fragmentos multimedia

En la Figura 3.16 se observa el código encargado de obtener la tabla de costo por atributo, contiene información sobre qué atributo, en qué sitio es más requerido mediante el cálculo que multiplica la veces que existe en una consulta por el peso del atributo por el tipo de operación (Select=1, Create=2, Delete=2 y Update=3), posteriormente se asigna a los sitios que tengan el valor más alto.

```
public ArrayList<TablaCosto> obtenerTablaCosto() {
        ArrayList<String> cad = this.arrCad;
        ArrayList<Query> tablafrecuencia = arrQuerys(cad);
        @SuppressWarnings("unchecked")
        ArrayList<TablaCosto> tabla = new ArrayList();
        tablafrecuencia.forEach(new Consumer<Query>() {

            public void accept(Query t) {
                for (int i = 0; i < t.getAtr().size(); i++) {
                    int vo = 0;
                    double f = 0;
                    int ta = 0;
                    atributo a = t.getAtr().get(i);
                    atributo id = obteneratributoid();
                    ArrayList<atributo> arrT = new ArrayList<>();
                    arrT.add(id);
                    TablaCosto costo = new TablaCosto();

                    if (t.getTipoOpera().contains("select")) {
                        vo = 1;
                    }
                    if (t.getTipoOpera().contains("create")) {
                        vo = 2;
                    }
                    if (t.getTipoOpera().contains("delete")) {
                        vo = 2;
                    }
                    if (t.getTipoOpera().contains("update")) {
                        vo = 3;
                    }
                    f = t.getFrecuencia();
                    ta = parseInt(a.getTamanio());
                    costo.setAtributo(a);
                    arrT.add(a);
                    costo.setArrAtri(arrT);
                    costo.setSitio(t.getSitio());
                    costo.setCosto((vo * f * ta));
                    tabla.add(costo);
                }
            }
        });

        return tabla;

    }
```

Figura 3.16 Creación de la tabla de costo

La estructura del tipo de datos *TablaCosto* contiene la información de la dirección del sitio donde se colocará el fragmento, el atributo más usado en ese sitio, los atributos que se asignarán a ese sitio mediante un arreglo, y permite solo tener un método de fragmentación y asignación. En la

Figura 3.17 se observa el método que fragmenta y asigna los atributos mediante un arreglo de *TablaCosto*, cada posición del arreglo contiene la información suficiente para esta labor.

```
 private void crearFragmento(TablaCosto frag) {
        try {
            String sql;

            AccesoDatosPG acc = new AccesoDatosPG(frag.getSitio().getIp() + ":" + "5432", this.nombd,
this.usubd, this.passbd);
            if (acc.conectar()) {

                sql = "drop table if exists " + frag.getNombre() + " ; CREATE TABLE " + frag.getNombre() + "
( " + atributosTipos(frag.getArrAtri()) + " );";

                acc.ejecutarComando(sql);
                acc.desconectar();
            }
            ArrayList atributos = null;
            ArrayList n = null;
            AccesoDatosPG acc2 = new AccesoDatosPG(this.dirbd + ":" + "5432", this.nombd, this.usubd,
this.passbd);
            if (acc2.conectar()) {
                String numero = "Select count (*) from " + this.getTabla().getNombre() + ";";
                n = acc2.ejecutarConsulta(numero);
                acc2.desconectar();
            }
            int num = ((Double) Double.parseDouble("" + ((ArrayList) n.get(0)).get(0))).intValue();
            int nLote = num / 50;
            System.out.println("Lotes**********************" + frag.getNombre());
            for (int i = 0; i < 50; i++) {
                acc2 = new AccesoDatosPG(this.dirbd + ":" + "5432", this.nombd, this.usubd, this.passbd);
                if (acc2.conectar()) {
                    if (i == 0) {
                        sql = "Select " + atri(frag.getArrAtri()) + " from " + this.getTabla().getNombre() +
" Limit " + nLote + " offset " + (i + 1);
                    } else {

                        sql = "Select " + atri(frag.getArrAtri()) + " from " + this.getTabla().getNombre() +
" Limit " + nLote + " offset " + nLote * (i + 1);
                    }
                    atributos = acc2.ejecutarConsulta(sql);
                    acc2.desconectar();
                }
                if (atributos != null) {
                    acc = new AccesoDatosPG(frag.getSitio().getIp() + ":" + "5432", this.nombd, this.usubd,
this.passbd);
                    if (acc.conectar()) {
                        for (int k = 0; k < atributos.size(); k++) {
                            String valor = "";
                            ArrayList tupla = (ArrayList) atributos.get(k);
                            for (int j = 0; j < tupla.size(); j++) {
                                valor += "'" + tupla.get(j) + "',";
                            }
                            valor = valor.substring(0, valor.length() - 1);
                            sql = "insert into " + frag.getNombre() + "(" + frag.getAtributos() + ") values
(" + valor + ");";

                            acc.ejecutarComando(sql);
                        }
                        acc.desconectar();
                    }
                }
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }
```

Figura 3.17 Método que fragmenta y asigna

3.4 Validación

Como parte de esta investigación se desarrolló un módulo para una aplicación existente nombrada XAMANA, desde la cual se aplica el método de fragmentación vertical para base de datos que considera consultas basada en contenido, la aplicación permite analizar un archivo de carga proporcionado para extraer la información indispensable para el método, conectarse al servidor remotamente, seleccionar una tabla de todo el esquema encontrado en el gestor de bases de datos, mostrar resultados preliminares del análisis y aplicar el esquema a lo largo de los sitios. En el capítulo 4 se aborda detalladamente la descripción de la aplicación Web.

Para validar el método implementado se presenta un caso de estudio y una comparación en el Capítulo 4. El caso de estudio se basa en fragmentar una base de datos multimedia simple que consta de una tabla que pertenece a una empresa que se dedica a la venta de maquinaria agrícola, la comparación se realiza con el trabajo elegido en el análisis de métodos de fragmentación vertical.

Capítulo 4. Resultados

En este capítulo se muestra el resultado del análisis de los trabajos seleccionados del estado del arte y se describe el funcionamiento de la aplicación Web.

4.1 Resultados del análisis

Para finalizar la etapa del análisis, en esta sección se presentan los resultados mediante gráficas y tablas. De esta manera se observa en las siguientes gráficas las comparaciones del análisis sobre los trabajos elegidos.

La Figura 4.1 muestra la comparación del número de artículos de cada editorial: la mayoría se concentra en IEEE; cinco artículos de esta editorial reúnen tres características de interés: ([22], [33], [44], [45] y [47]), pero no consideran datos multimedia; algunos métodos son completos ([32], [33], [42], [44], [45], [47]) y/o fáciles de implementar ([22], [33], [41], [47]), y pocos tienen un modelo de costos ([22], [32], [41]–[43], [45], [47]–[49]). En cuanto a ACM, solo se presentaron tres artículos ([23], [27], [40]), que tienen en cuenta bases de datos relacionales o big data. Springer publicó el artículo que contiene la mayoría de características: fragmentación vertical, completitud y facilidad implementación, incluye un modelo de costos y está orientado a bases de datos multimedia ([13]), pero no considera CBIR. Con respecto a Elsevier, se centran en la fragmentación vertical de las bases de datos relacionales ([18], [19], [28], [50], [51]). Finalmente, en otras editoriales, hay enfoques para CBIR ([30], [31], [57]–[59]), también hay un artículo ([34]) que cumple con casi todas las características, pero está orientado a la fragmentación horizontal.

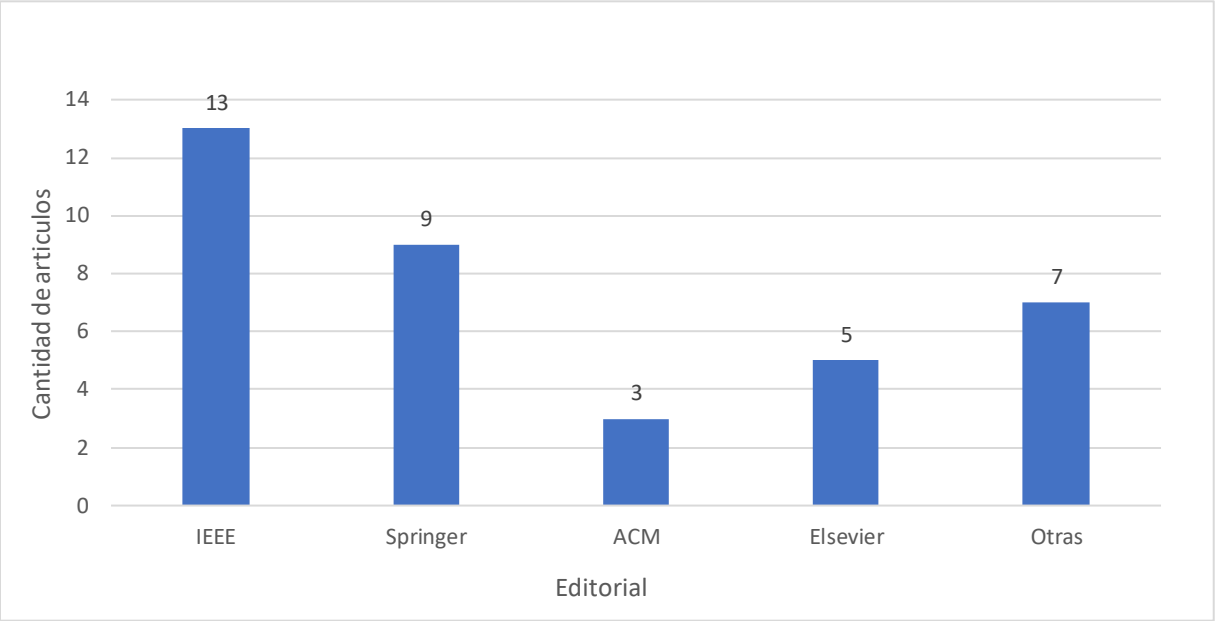


Figura 4.1 Artículos por editorial

La gráfica de la Figura 4.2 muestra una comparativa de los artículos que presentan toda la información necesaria para replicar el método propuesto, se encontraron 24 de 37 artículos con esta característica ([13], [23]–[25], [28], [30]–[34], [42], [44]–[47], [49], [51], [53], [54], [56], [58]–[60]). El resto no cuenta con la información necesaria para su implementación, faltan datos como el entorno o sistema gestor de la base de datos en la que fueron desarrollados o la explicación de un alguno de sus pasos ([18], [19], [22], [26], [29], [40], [41], [43], [48], [50], [52], [55], [57]).

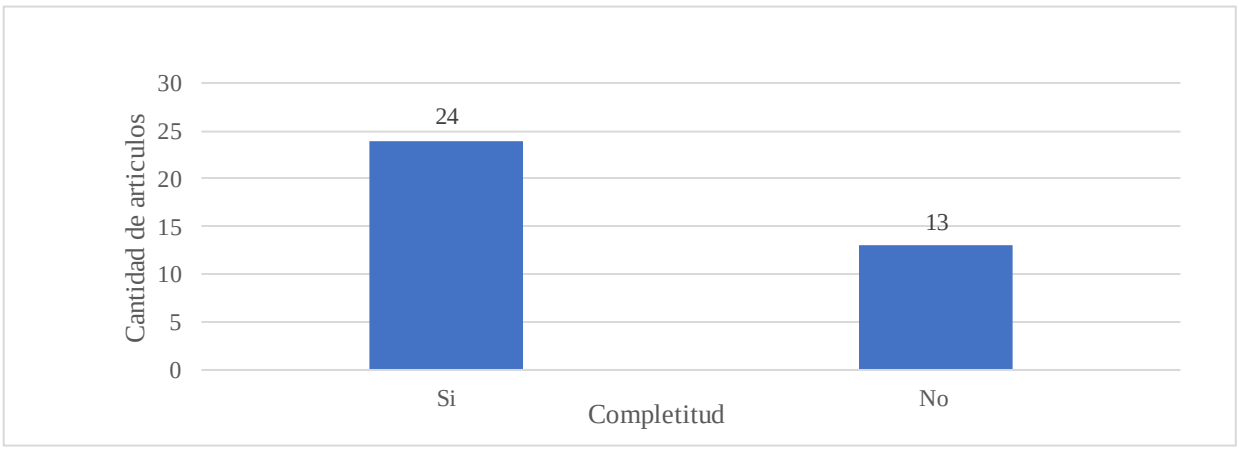


Figura 4.2 Artículos con la información completa

La Figura 4.3 muestra que la mayoría de los métodos no presentan un modelo de costos, solo 21 de los 37 lo incluyen, la mayoría fueron publicados en IEEE con 9 artículos ([22], [32], [41]–[43], [45], [47]–[49]), mientras que Elsevier, ACM, Springer y otras editoriales tienen 4 ([19], [28], [50], [51]), 2 ([27], [40]), 4 ([13], [24], [26], [55]) y 2 ([34], [60]) propuestas, respectivamente. Algunos modelos de costos consideran E/S de disco ([22], [24], [41], [50]) o el acceso a atributos irrelevantes ([13], [43], [45], [47], [55]), mientras que otros también incluyen el costo de transporte ([19], [26]–[28], [32], [34], [40], [42], [48], [49], [51], [60]).

La Figura 4.4 muestra un gráfico relacionado con el número de artículos que tienen en cuenta algún método para la recuperación basada en contenido; 7 de 37 propusieron un método CBIR. Aunque [13] y [34] contienen casi todas las características de nuestra metodología de análisis, solo una propuesta [34] involucra consultas basadas en contenido, pero se centra en la fragmentación horizontal. En [57], se propone un algoritmo de selección de características para la clasificación de imágenes hiperespectrales, el índice de similitud se combina con el algoritmo de agrupamiento de k-medias para obtener bandas importantes. El descriptor SURF (característica robusta acelerada) se utiliza en [30], [34], [46]. En [59], se extraen las características de Haralick, que también se conoce como matriz de co-ocurrencia de nivel de gris (GLCM) junto con características de patrón binario local (LBP) e histograma de gradientes orientados (HOG).

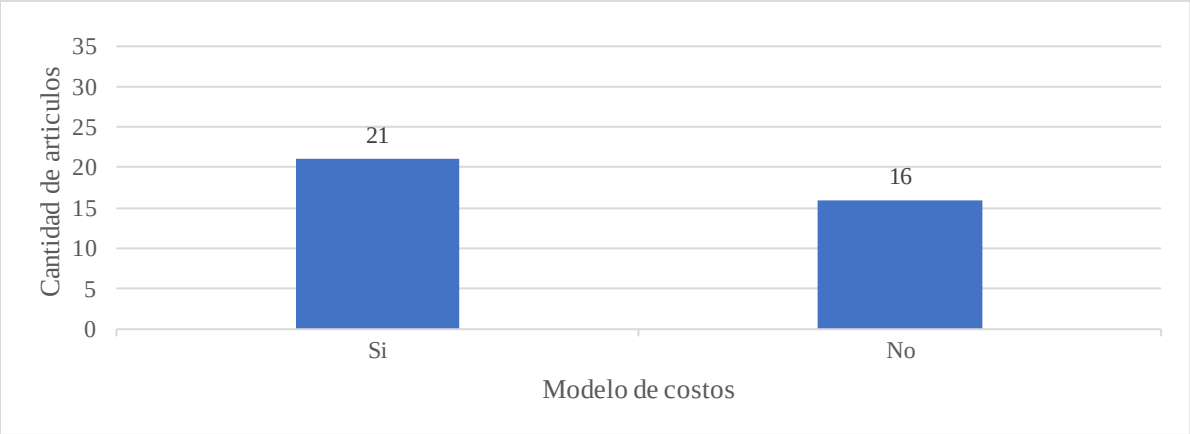


Figura 4.3 Artículos que contiene un modelo de costo

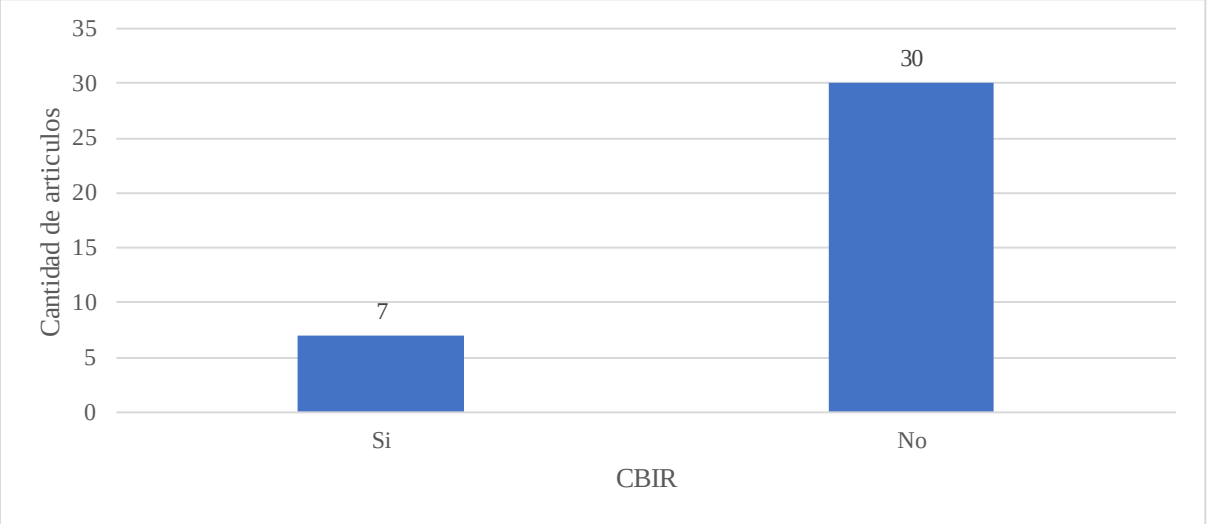


Figura 4.4 Artículos CBIR

La gráfica de la Figura 4.5 muestra una comparación de los artículos que contienen una implementación fácil, es decir, presentan una explicación clara de los pasos de sus propuestas, incluyendo ejemplos e información sobre las tecnologías utilizadas para su desarrollo. Se considera que 11 de 37 tienen alta facilidad de implementación [13], [18], [19], [22], [23], [30], [33], [34], [41], [51], [53], el resto de artículos no cuenta con la información completa para su implementación o presentan una alta dificultad para su replicación [24]–[29], [31], [32], [40], [42]–[46], [48]–[50], [52], [54]–[60].

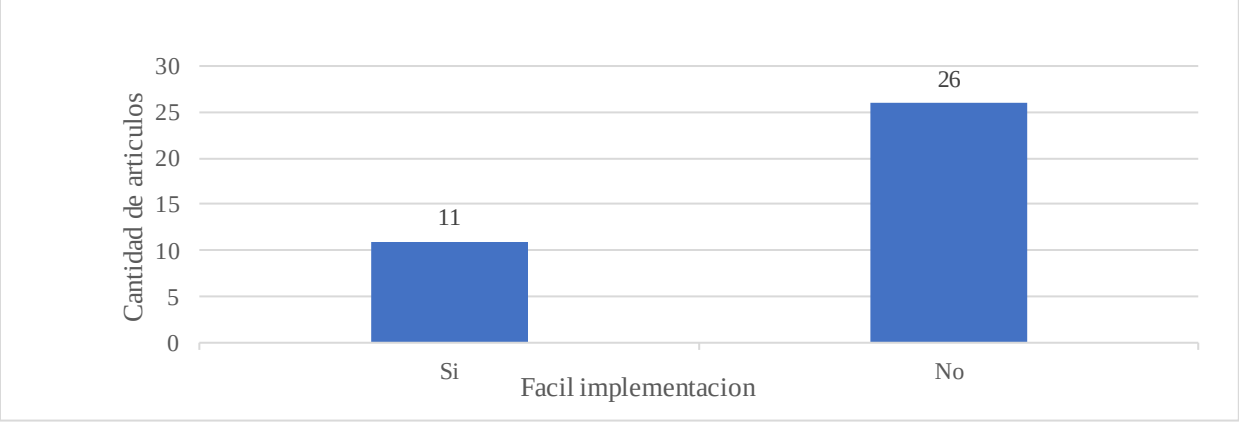


Figura 4.5 Artículos con fácil implementación

La Figura 4.6 presenta una gráfica relacionada con el número de artículos donde se implementaron diferentes tipos de bases de datos, el tipo de base de datos distribuida fue el considerado en más artículos [18], [19], [26], [27], [28, p.], [29], [40]–[42], [44], [48], [50], [51], [55], [56], [60], se aplicaron 9 propuestas en bases de datos multimedia [13], [30], [31], [34], [46], [52], [57]–[59], 5 en relacionales [24], [32], [33], [53], [54] y 7 en otro tipo [22], [23], [25], [43], [45], [47], [49]. La Figura 4.7 muestra el sistema gestor de bases de datos (SGBD) utilizado por los métodos de fragmentación horizontal, la mayoría de los artículos no lo especifican [13], [18], [19], [26]–[29], [31]–[33], [40]–[44], [46]–[48], [50], [51], [54]–[57], [59], [60]. Los SGBD utilizados por los enfoques analizados son PostgreSQL [23], [30], [52], [53], Hive [22], [25], HBase [45], H-Store [49], Oracle [24], MongoDB [34] y Corel Paradox [58].

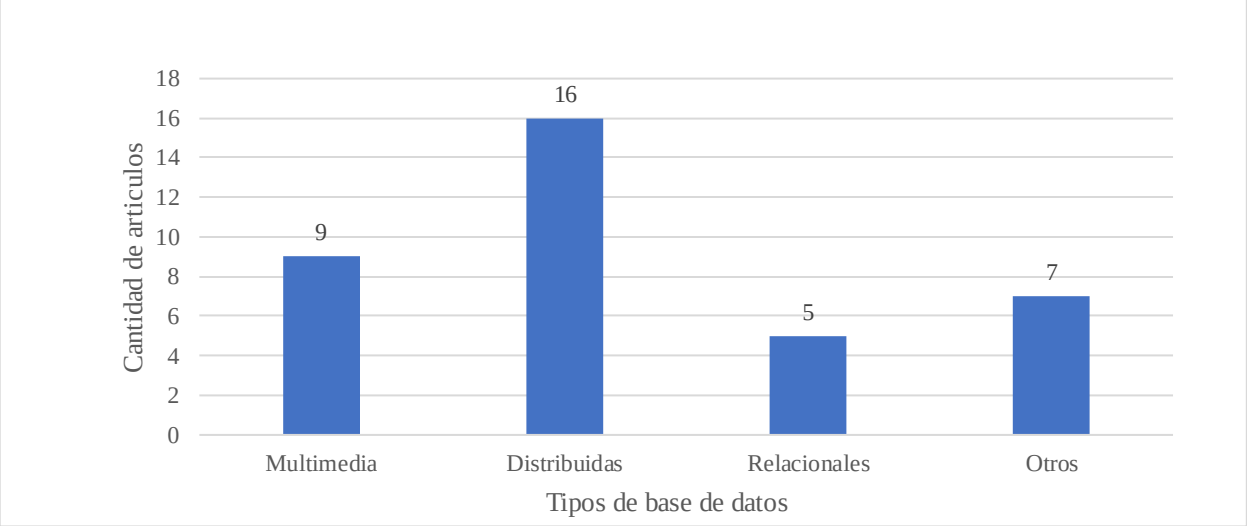


Figura 4.6 Tipos de base de datos

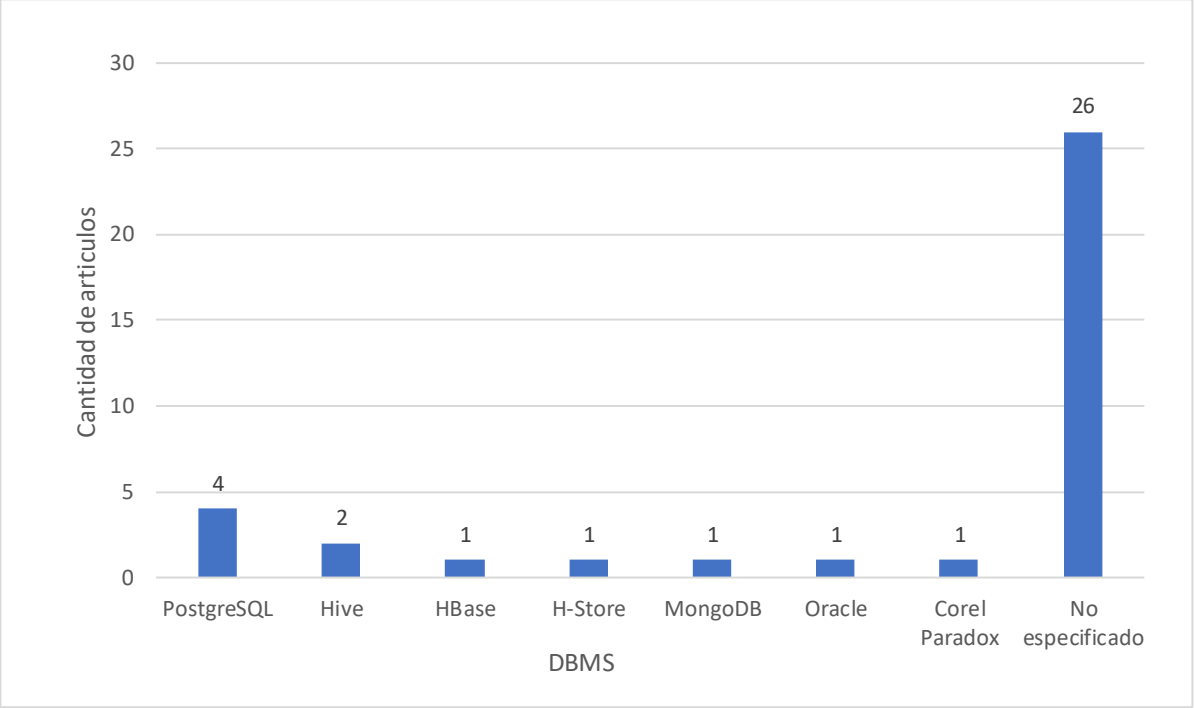


Figura 4.7 Gestores de bases de datos

4.2 Aplicación Web

Como inicio de la aplicación XAMANA se muestra en la pantalla de inicio un formulario donde se permite la elección de un gestor de bases de datos como se muestra en la Figura 4.8, cuando el usuario administrador haga clic en el menú desplegable, se presentarán los gestores de colecciones de datos y de bases de datos, ya que el sistema está pensado para soportar varios sistemas gestores de bases de datos, estas opciones son MySQL, Postgres-XL, PostgreSQL y MongoDB. Este proyecto se centra en las bases de datos que derivan de PostgreSQL.

A continuación, se muestra en la Figura 4.8 la pantalla de inicio de la aplicación web.

Figura 4.8 Inicio XAMANA

El usuario entonces elige del menú el gestor de bases de datos al que se quiere conectar, en este caso como se observa en la Figura 4.9 se selecciona Postgres-XL, después de seleccionar el SGBD, aparece un botón para la conexión, como se muestra en la Figura 4.10.

Figura 4.9 Selección del gestor de bases de datos

En la Figura 4.10 se muestra cuando el usuario administrador de base de datos seleccionó la opción deseada, en ese momento hace clic en el botón para continuar con la configuración de la conexión.

Figura 4.10 Elección Postgres-XL

Continuando con la configuración de conexión, se observa en la Figura 4.11, un formulario en el cual el usuario debe ingresar la dirección IP, el puerto y el nombre de la base de datos, así como el usuario administrador de la base de datos y la contraseña del usuario. Cuando existe una conexión a esa base de datos se despliegan las opciones de las tablas existentes en esa base de datos.

Figura 4.11 Formulario de conexión

Se observa en la Figura 4.12 que, al ingresar estos datos, las opciones de las tablas son tres, se elige la tabla que se desea fragmentar. Después de que el usuario ingresó los datos y eligió la tabla, es posible seleccionar la opción de cancelar si lo desea o la opción de conexión de prueba, para continuar. Si no hubiera conexión, en la opción donde se eligen las tablas no aparecería ninguna tabla para elegir.

Figura 4.12 Llenado correcto de conexión

Después de haber ingresado los datos correctos como se muestra en la Figura 4.13, se debe hacer clic en el botón de la conexión de prueba. Al tener una conexión correcta, el sistema presenta un mensaje en el cual informa que la conexión fue exitosa y el siguiente paso es configurar la fragmentación, el usuario debe presionar el botón *Configurar la fragmentación* para seguir con el formulario que permite dicha configuración como se muestra en la Figura 4.14.

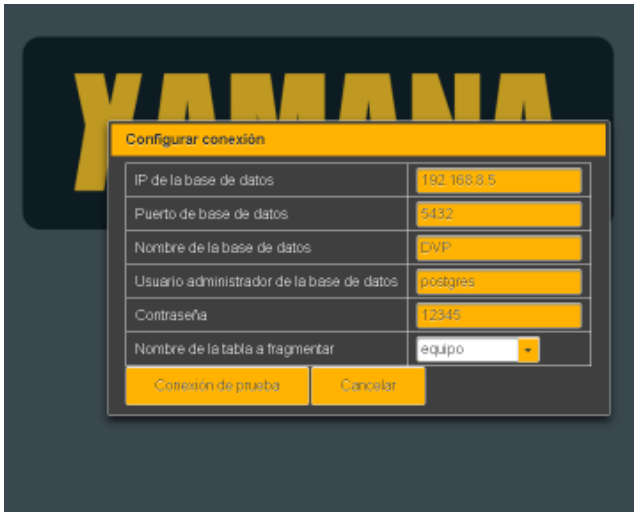


Figura 4.13 Conexión a la base de datos

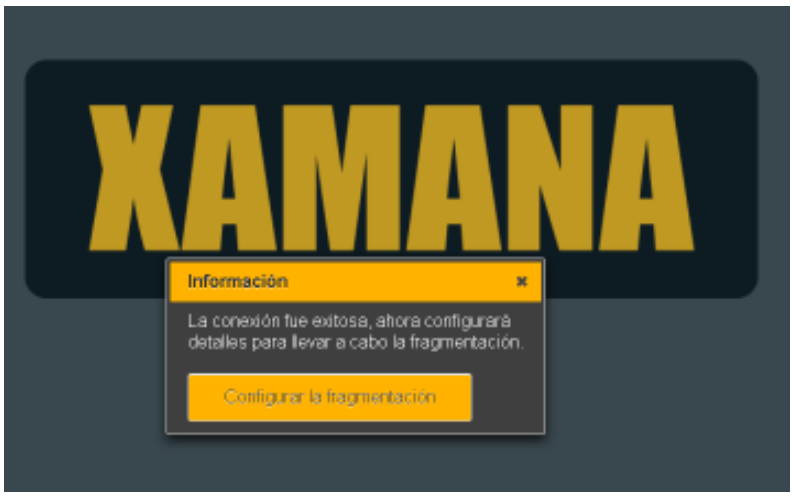


Figura 4.14 Alerta de conexión

En la Figura 4.15 se muestra un formulario de configuración, el cual permite la selección del tipo de fragmentación, entre estas opciones se encuentra la fragmentación horizontal, la vertical y la híbrida. El siguiente apartado es posible elegir la opción cuando existen columnas que involucran descriptores o datos multimedia, por defecto esta opción aparece en "No", si en la tabla a fragmentar existieran atributos multimedia y atributos descriptores, al hacer clic en el botón se establecen los atributos multimedia y sus descriptores. En el apartado de archivo de registro al presionar el botón llamado *Escoger*, se abre el gestor de archivos del sistema operativo que permite elegir el archivo de registro de la base de datos. En las opciones, *Umbral de operaciones* y *Umbral de costo*, es posible elegir entre valores del menos cien al cien positivo, y al tener las opciones de formulario llenas, el usuario administrador de base de datos debe hacer clic en el botón *Generar esquema*.

Figura 4.15 Formulario de configuración de la fragmentación

En la Figura 4.16 se observa que el usuario eligió fragmentación vertical. En la Figura 4.17, al seleccionar "Sí" en la opción CBIR, se abre un recuadro donde se establecen los atributos multimedia y descriptores en pares.

Figura 4.16 Elección de tipo de fragmentación

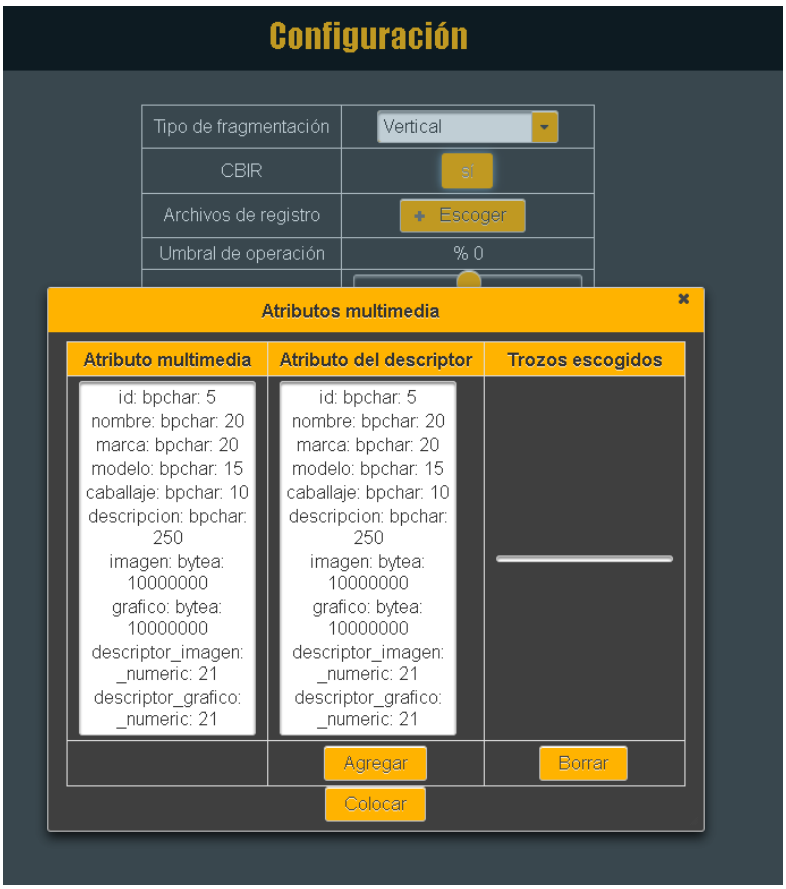


Figura 4.17 Atributos multimedia

Para la creación de pares multimedia, el usuario debe elegir en la columna de atributos multimedia el atributo de la tabla que sea multimedia, y en la columna de atributo del descriptor se elige el descriptor visual del atributo elegido en la columna atributo multimedia, a continuación, el usuario debe hacer clic en el botón *Agregar*, el cual permite colocar los pares en una nueva columna para que sean enviados para su ubicación en un solo fragmento. Si existieran más atributos multimedia y en la misma tabla estuvieran los atributos que son sus descriptores, se deben elegir de la misma manera. En las Figuras 4.18 y 4.19 se observa que ya fueron seleccionados pares para la fragmentacion, el usuario debe presionar el boton *Colocar* para terminar con la eleccion de atributos multimedia y sus atributos descriptores.

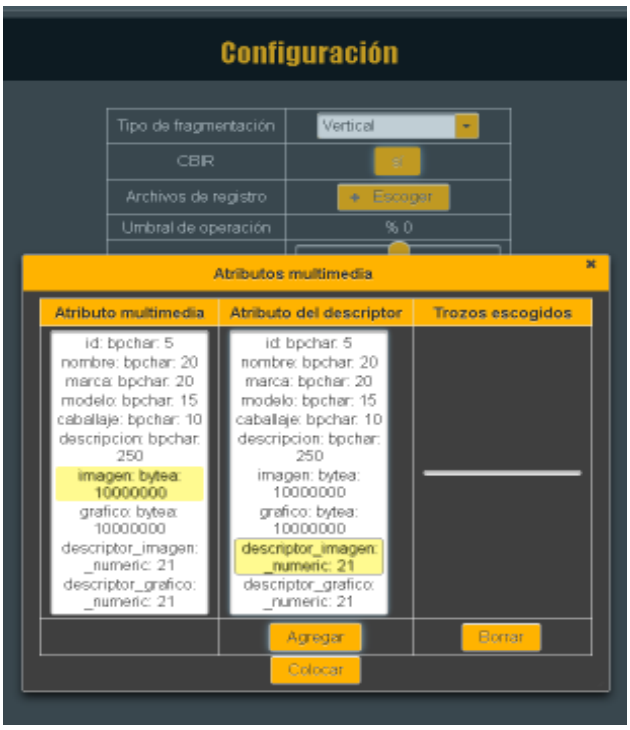


Figura 4.18 Elección de atributos multimedia y atributos descriptores

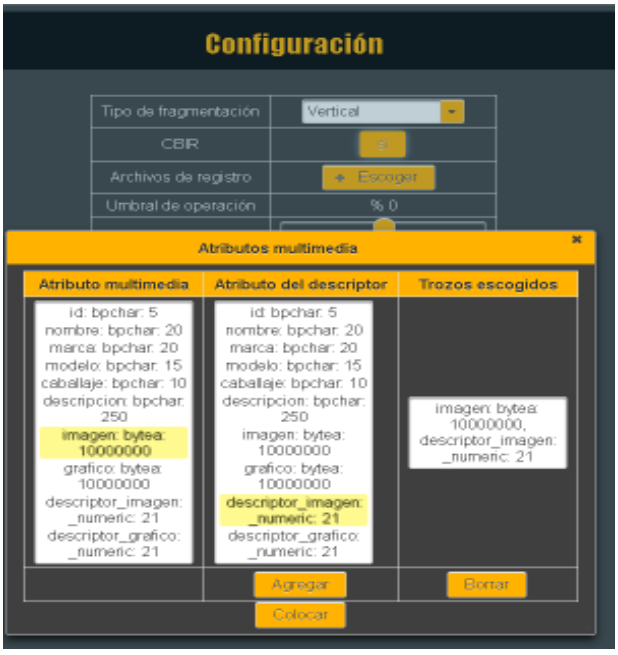


Figura 4.19 Fragmentos multimedia elegidos

La siguiente opción permite elegir el archivo de registro, al hacer clic en el botón *Escoger* se abre una ventana del gestor de archivos del sistema operativo, se busca el archivo y se elige para su carga, cuando se haya seleccionado (Figura 4.20), se observa que la opción de escoger sigue activa, lo que permite elegir más archivos de registro para su análisis, en la Figura 4.21.

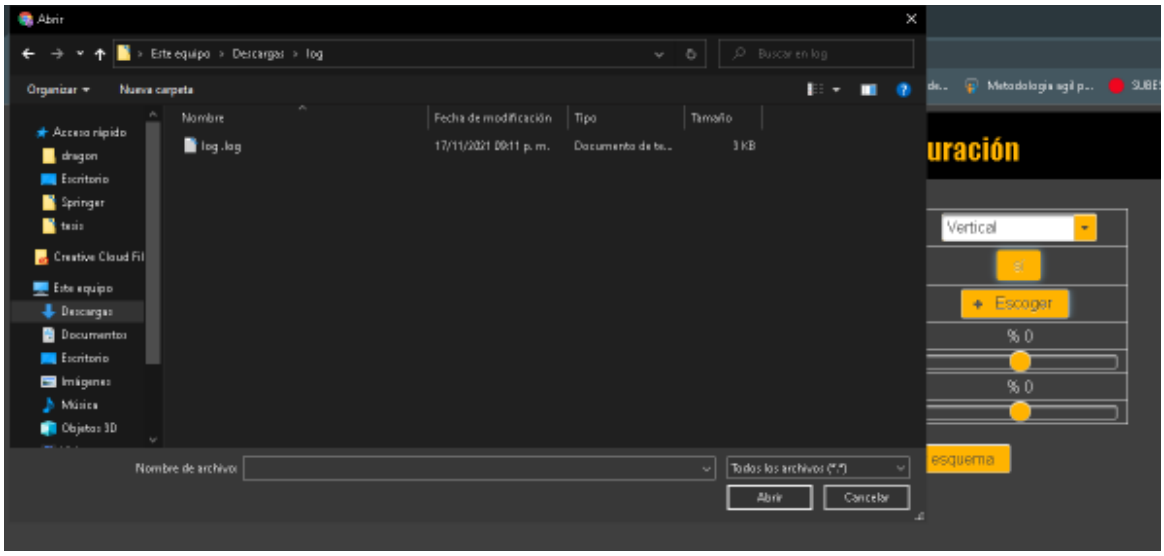


Figura 4.20 elección del archivo de registro

Figura 4.21 Opción de agregar más archivos de registro

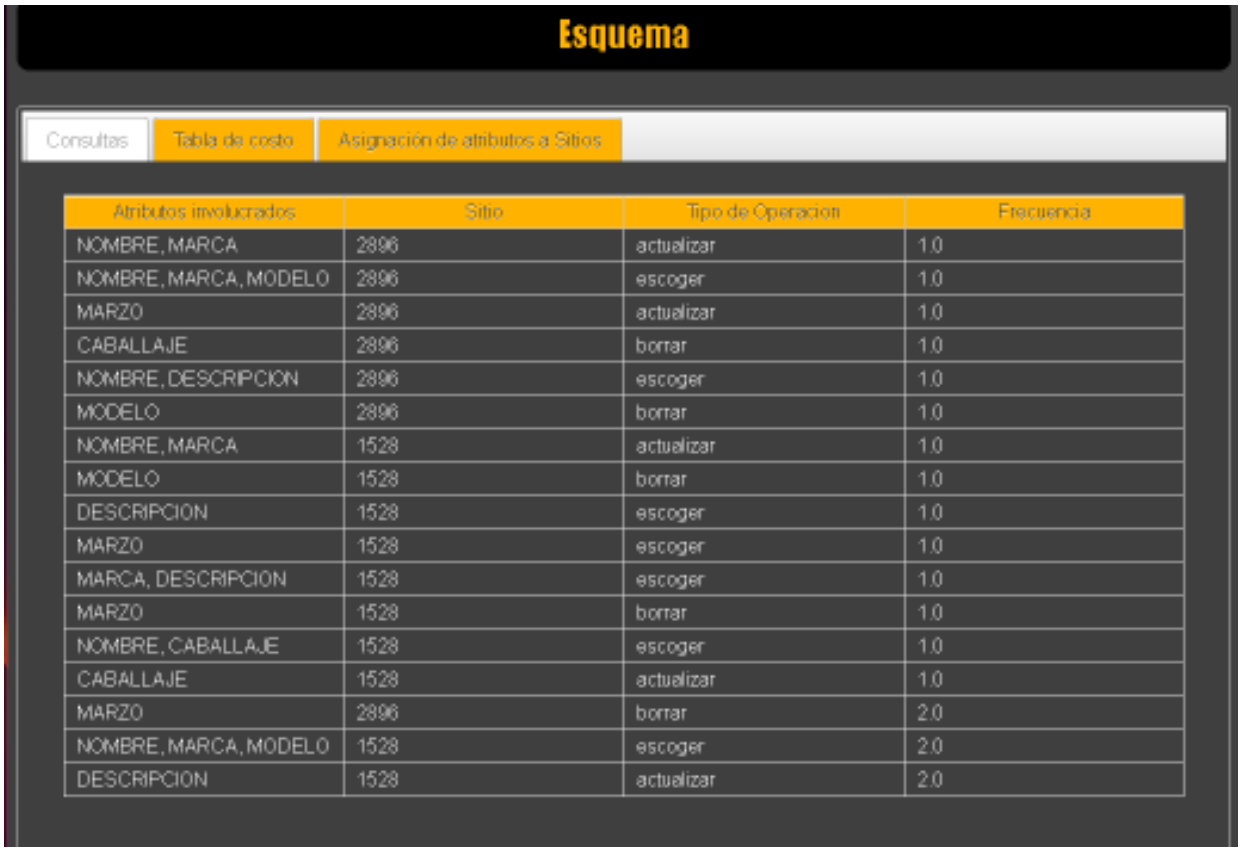
El método desarrollado está enfocado en la fragmentación vertical para base de datos multimedia que considera consultas basadas en contenido, pero es estático, si fuera dinámico, se elegiría algún valor de las opciones de los umbrales, en este caso de dejan en el valor 0, como se muestra en la Figura 4.21.

Al término del llenado del formulario de la configuración de fragmentación, el usuario debe hacer clic en el botón *Generar esquema*, entonces comienza el proceso del análisis del archivo de registro mientras que en la pantalla se encuentra una animación de carga (Figura 4.22).

Figura 4.22 En proceso del análisis

Después del término de este proceso se muestra el resultado en pantalla, se encuentra una tabla en la que se presentan las consultas existentes en el archivo de registro de la base de datos, cada fila existente en esa tabla corresponde a una consulta del archivo de registro. La primera columna muestra los nombres de los atributos que se involucran en esa consulta, la columna llamada *Sitio*, proporciona el nombre del sitio en el que fue realizada esa consulta. La columna con el nombre *Tipo de operación*, muestra la clase de operación que se realizó en esa consulta, en la columna *Frecuencia* se encuentra el número de veces que se realizó esa consulta en el mismo sitio, que

involucra los mismos atributos, y con la misma operación. En la Figura 4.23, se observa con detalle lo descrito anteriormente.



Atributos involucrados	Sitio	Tipo de Operacion	Frecuencia
NOMBRE, MARCA	2896	actualizar	1.0
NOMBRE, MARCA, MODELO	2896	escoger	1.0
MARZO	2896	actualizar	1.0
CABALLAJE	2896	borrar	1.0
NOMBRE, DESCRIPCION	2896	escoger	1.0
MODELO	2896	borrar	1.0
NOMBRE, MARCA	1528	actualizar	1.0
MODELO	1528	borrar	1.0
DESCRIPCION	1528	escoger	1.0
MARZO	1528	escoger	1.0
MARCA, DESCRIPCION	1528	escoger	1.0
MARZO	1528	borrar	1.0
NOMBRE, CABALLAJE	1528	escoger	1.0
CABALLAJE	1528	actualizar	1.0
MARZO	2896	borrar	2.0
NOMBRE, MARCA, MODELO	1528	escoger	2.0
DESCRIPCION	1528	actualizar	2.0

Figura 4.23 Resultado de las consultas existentes

También se encuentra una pestaña llamada *Tabla de costo*, en la cual se muestra una tabla que contiene la información de los costos por atributo calculados mediante el modelo de costos por atributo, en la tabla se observa el sitio en el que cada atributo fue más costoso, en la columna llamada *Sitio* se encuentran las direcciones IP, estas corresponden a los sitios donde se realizaron las consultas, en la siguiente columna se observa el nombre del atributo que fue más requerido en ese sitio y en la columna *Costo* se muestra el valor por el cual se asignó a ese sitio. En la Figura 4.24 se observa lo descrito anteriormente.

Figura 4.24 Tabla de costo por atributo

En la pestaña llamada *Asignación de atributos a sitios*, se encuentra una tabla que describe la formación de fragmentos, en la columna *Nombre de Frag.* se observan los nombres de los fragmentos a crear, en la columna *Sitio* se encuentra la dirección IP del nodo de la red donde se colocará cada fragmento y en la columna *Atributos* se encuentran los nombres de los atributos que se asignan a cada fragmento (Figura 4.25). Para proceder con la fragmentación y asignación a sitios, el usuario debe de hacer clic en el botón *Hagámoslo*, al ejecutar la orden, el sistema muestra una barra de progreso que va desde el 1 al 100 de acuerdo con el porcentaje de la fragmentación. Se observa en las figuras 4.26 y 4.27 cómo la barra de progreso se llena.

Figura 4.25 Asignación de fragmentos a sitios

Figura 4.26 Barra de progreso 26%

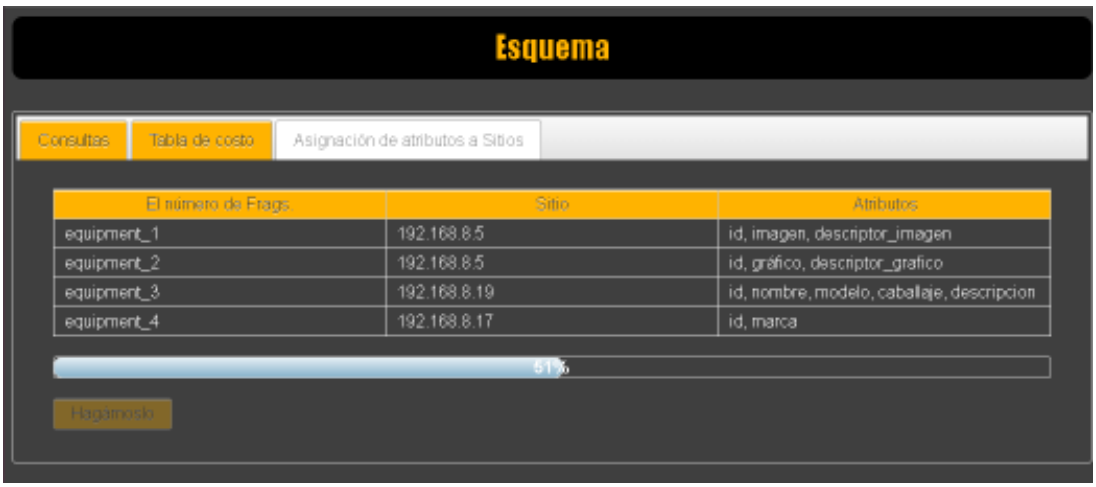


Figura 4.27 Barra de progreso 51%

Al término de la fragmentación se muestra un mensaje que informa que la fragmentación se realizó correctamente (Figura 4.28).

Figura 4.28 Fragmentación realizada

4.3 Demostración del método de fragmentación vertical

Como demostración se consideraron 3 sitios con las siguientes direcciones IP, sitio 1: 192.168.8.5, sitio 2: 192.168.8.17 y sitio 3: 192.168.8.19, y un archivo de registro con 19 consultas. En la Figura 4.29 se observan dos pantallas de inicio que pertenecen al sistema operativo Ubuntu que fueron instalados en el programa VirtualBox que permite tener máquinas virtuales en el equipo.

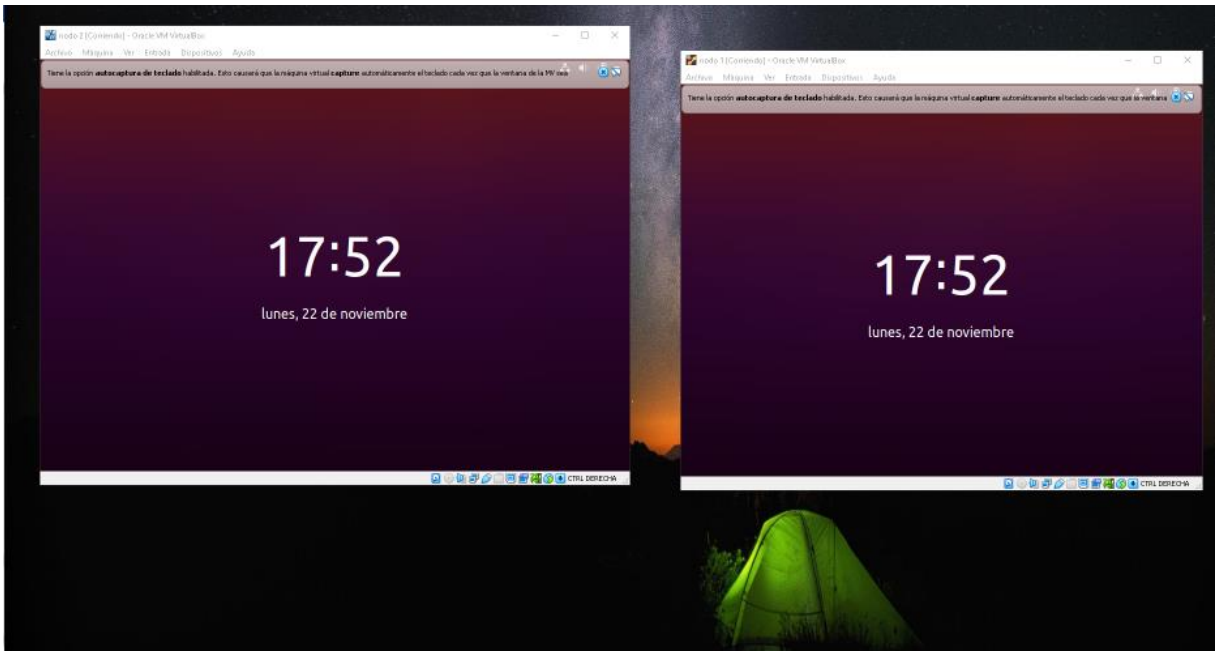


Figura 4.29 Sistemas operativos Ubuntu

En ellas se instaló PostgreSQL y para la administración, en la máquina física se instaló pgAdmin 4 que permite conectarse a las bases de datos de las máquinas virtuales. Se observa en la Figura 4.30 que las conexiones a las bases de datos están realizadas.

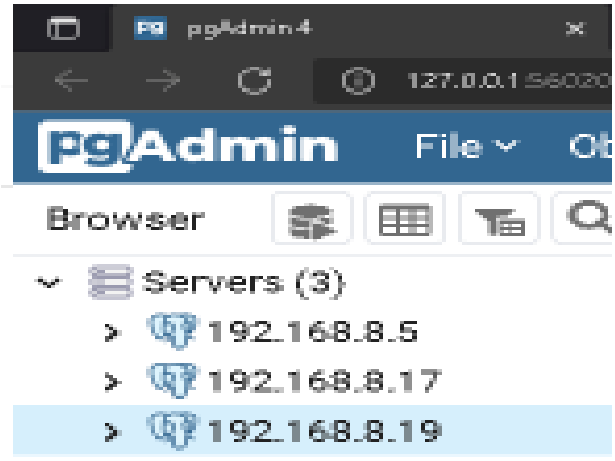


Figura 4.30 Conexión a las bases de datos

Para esta demostración se observa que, en cada nodo, no existen tablas creadas excepto en el nodo principal, el cual contiene la base de datos en la que se encuentra la tabla a fragmentar (Figuras 4.31, 4.32, 4.33).

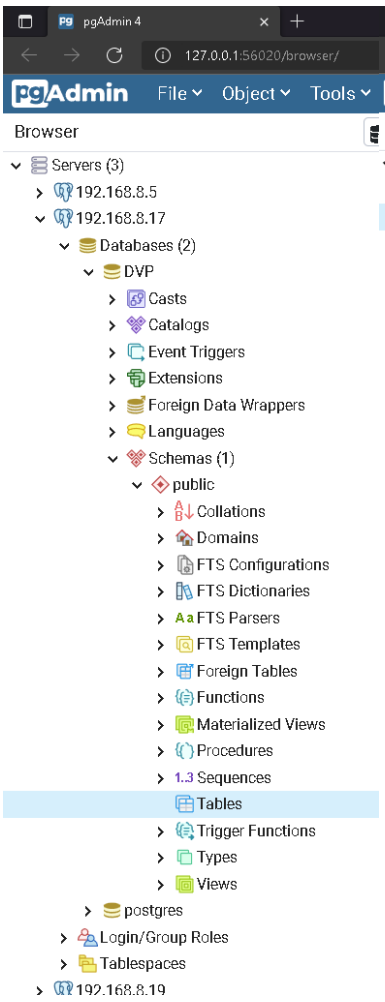


Figura 4.31 192.168.8.17 Vacío

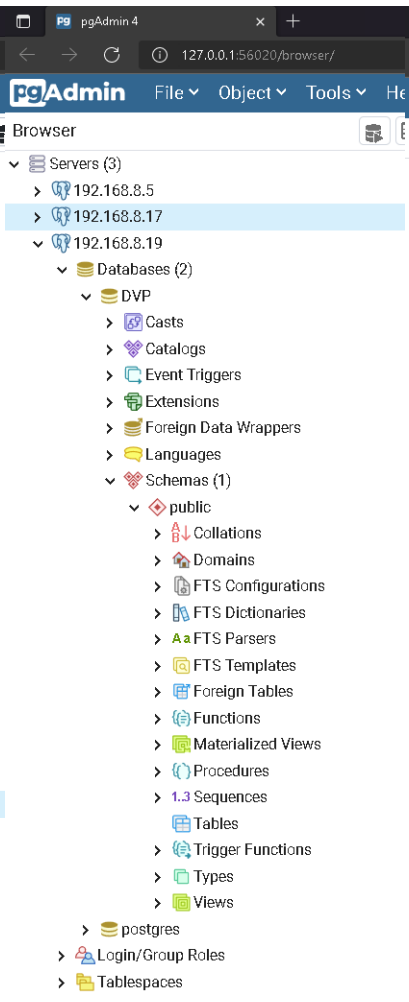


Figura 4.32 192.168.8.19 Vacío

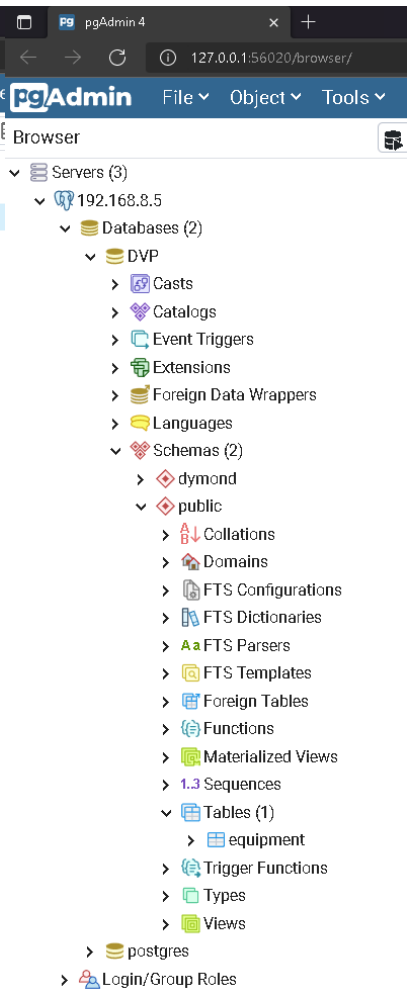


Figura 4.33 192.168.8.5 Tabla a fragmentar

Se utilizó el escenario de una base de datos multimedia simple usada para administrar equipo en una compañía de venta de maquinaria. La base de datos consiste en la tabla equipment (*id, nombre, marca, modelo, caballaje, descripción, imagen, descriptor_imagen, grafico, descriptor_grafico*). Para iniciar con la fragmentación se ingresa a la aplicación, se elige el gestor de bases de datos al que se quiere conectar como se muestra en la Figura 4.34, a continuación, se ingresan los datos correctos para la conexión a la base de datos lo que permite elegir la tabla que se fragmentará (Figura 4.35).

Figura 4.34 Selección de gestor

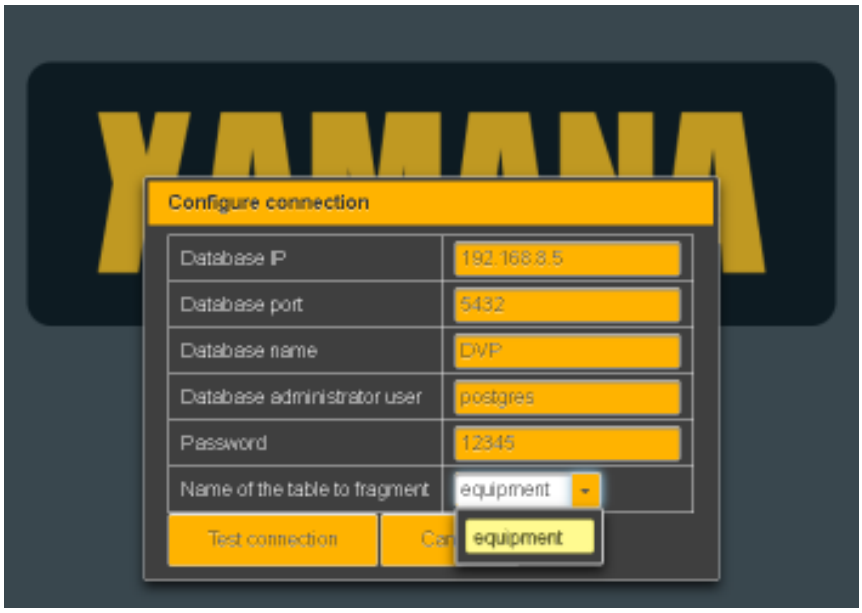


Figura 4.35 Datos de conexión

Después de la conexión exitosa se configura la fragmentación (Figura 4.36), eligiendo la opción de fragmentación vertical, se seleccionan los atributos que involucran descriptores (Figura 4.37), para este caso se establecieron los atributos *imagen* (atributo multimedia), *descriptor_imagen*

(atributo descriptor de la imagen), *grafico* (atributo multimedia) y *descriptor_grafico* (atributo descriptor del atributo gráfico). Se eligió el archivo de registro de la base de datos (Figura 4.38).

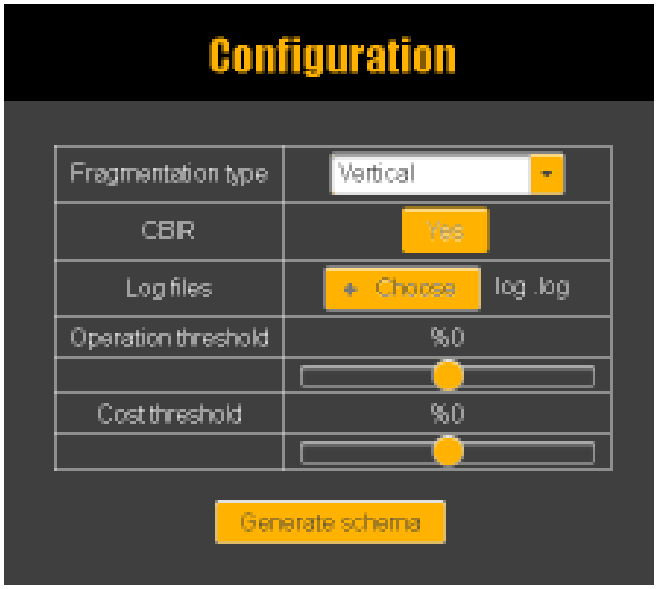


Figura 4.36 Elección de fragmentación vertical

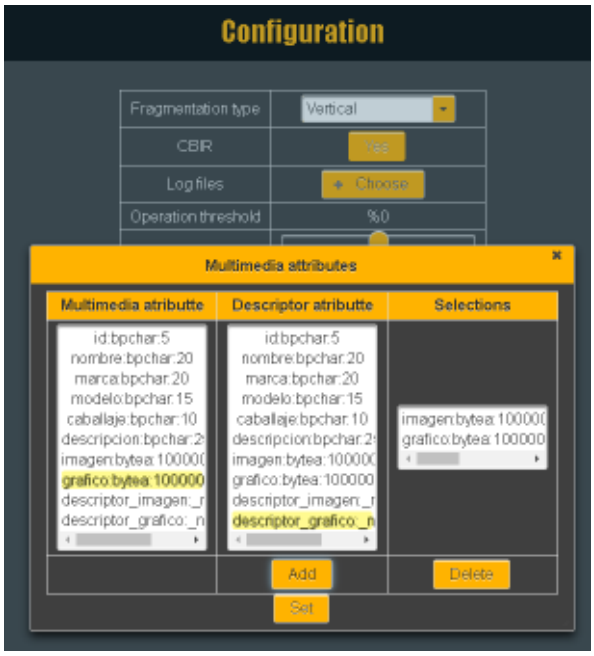


Figura 4.37 Elección de atributos multimedia-descriptor

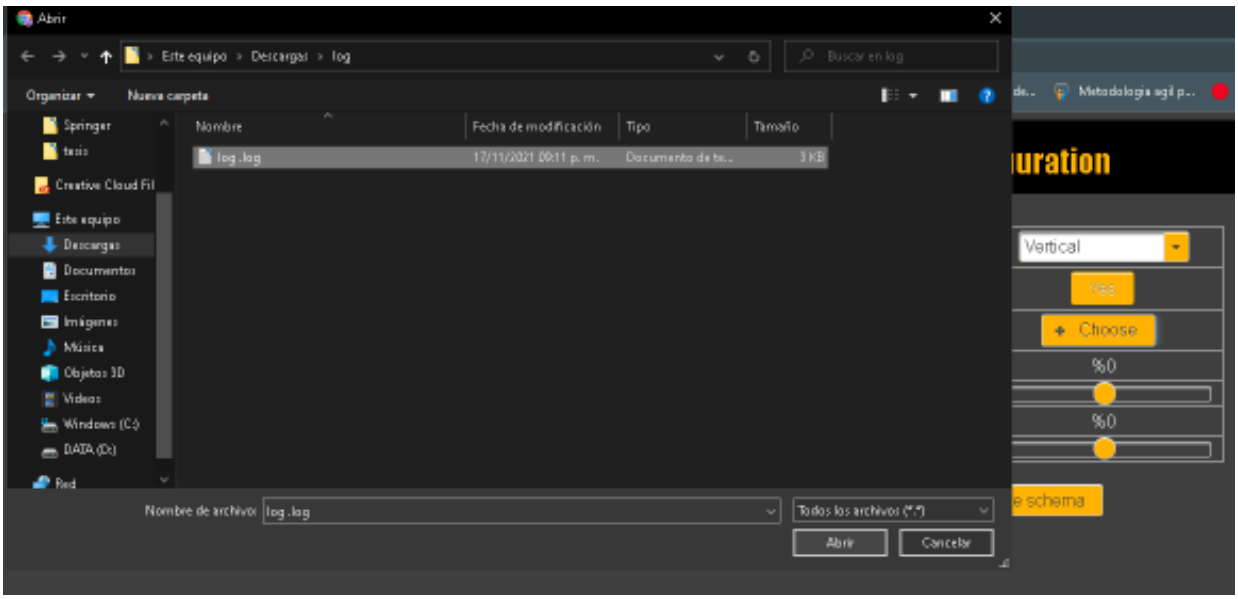
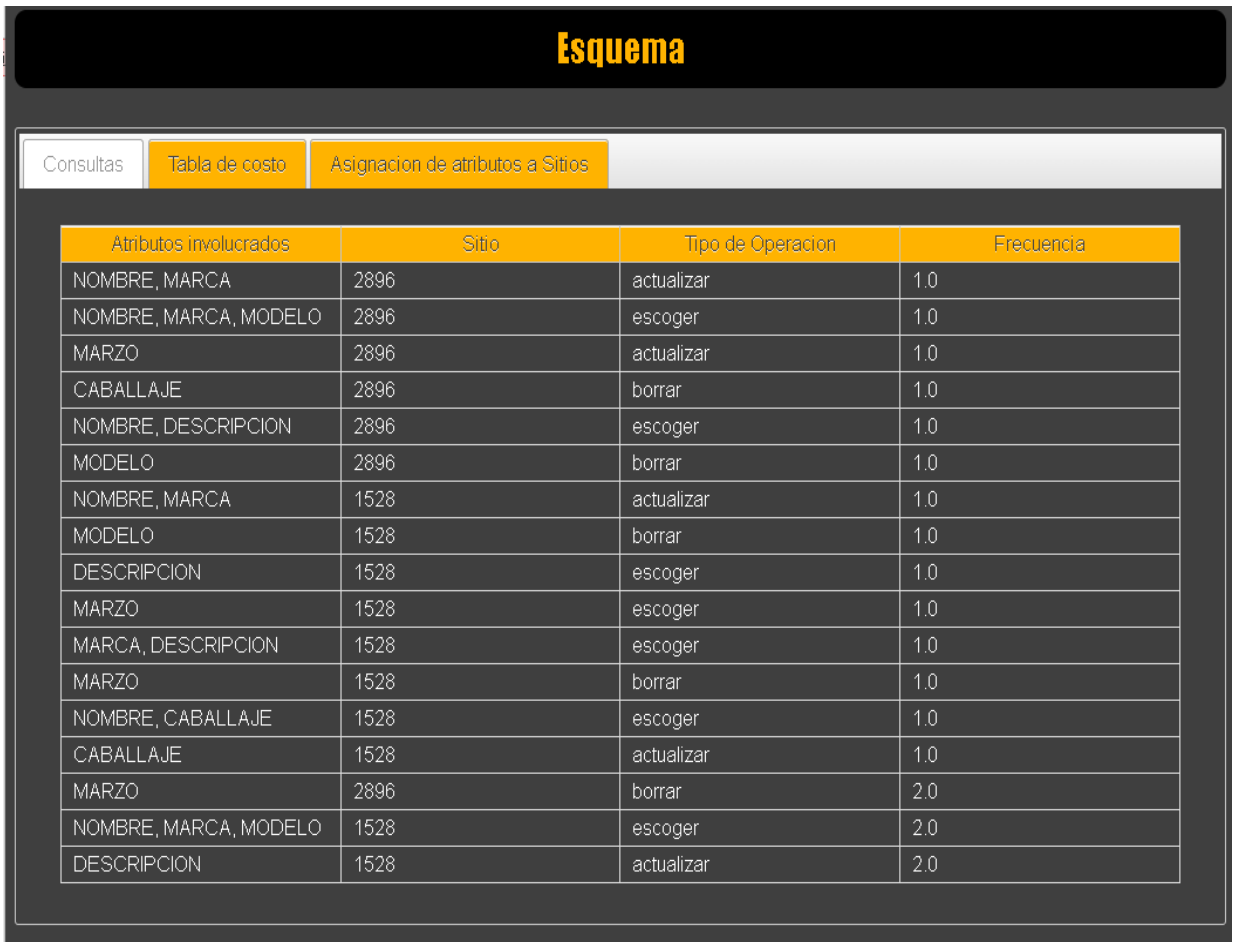


Figura 4.38 Elección del archivo de configuración

En las siguientes imágenes se observa la información extraída del archivo de registro de la base de datos que permite realizar los cálculos para obtener la tabla de costo para la colocación de atributos en los sitios donde más se requieren. La Figura 4.39 muestra la información de consultas existentes en el archivo de registro, en la Figura 4.40 se observa en qué sitio es asignado cada atributo y el costo que se obtuvo y, en la Figura 4.41 se visualiza cómo queda el esquema de fragmentación, se observa el nombre que recibe ese fragmento, la dirección del nodo en donde se colocará y los atributos que estarán en él.



Atributos involucrados	Sitio	Tipo de Operacion	Frecuencia
NOMBRE, MARCA	2896	actualizar	1.0
NOMBRE, MARCA, MODELO	2896	escoger	1.0
MARZO	2896	actualizar	1.0
CABALLAJE	2896	borrar	1.0
NOMBRE, DESCRIPCION	2896	escoger	1.0
MODELO	2896	borrar	1.0
NOMBRE, MARCA	1528	actualizar	1.0
MODELO	1528	borrar	1.0
DESCRIPCION	1528	escoger	1.0
MARZO	1528	escoger	1.0
MARCA, DESCRIPCION	1528	escoger	1.0
MARZO	1528	borrar	1.0
NOMBRE, CABALLAJE	1528	escoger	1.0
CABALLAJE	1528	actualizar	1.0
MARZO	2896	borrar	2.0
NOMBRE, MARCA, MODELO	1528	escoger	2.0
DESCRIPCION	1528	actualizar	2.0

Figura 4.39 Consultas analizadas del archivo de registro

Figura 4.40 Tabla de costo del caso de estudio

Figura 4.41 Esquema final de fragmentación

En la Figura 4.41 se observa que los atributos seleccionados anteriormente en la configuración están colocados en un fragmento junto con el atributo *id*, mientras que los atributos asignados a un mismo sitio en la sección anterior se colocan en un mismo fragmento que se asignó a ese nodo de la red. Al ordenar la fragmentación, la barra de progreso comienza a llenarse y cuando llega al

100 por ciento, indica que la fragmentación se ha realizado correctamente como se observa en la Figura 4.42.

Figura 4.42 Fragmentación de la tabla DVP realizada

Para verificar que se hayan creado los fragmentos, se ingresó a las bases de datos mediante pgAdmin 4, los fragmentos correspondientes a cada sitio fueron colocados correctamente, quedando de la siguiente manera la fragmentación, en el sitio 192.168.8.5, se colocaron los fragmentos multimedia **equipment_1**(*id, imagen, descriptor_imagen*) y **equipment_2**(*id, grafico, descriptor_grafico*), mientras que en el sitio 192.168.8.19 se habían asignado los atributos *nombre, modelo, caballaje y descripción*, como se asignaron al mismo sitio, quedaron en el mismo fragmento junto al *id*, finalmente, en el sitio 192.168.8.17 se asignó el fragmento **equipment_4**(*id, marca*).

Las Figuras 4.43, 4.44, 4.45 y 4.46 permiten verificar que están correctamente colocados los fragmentos.

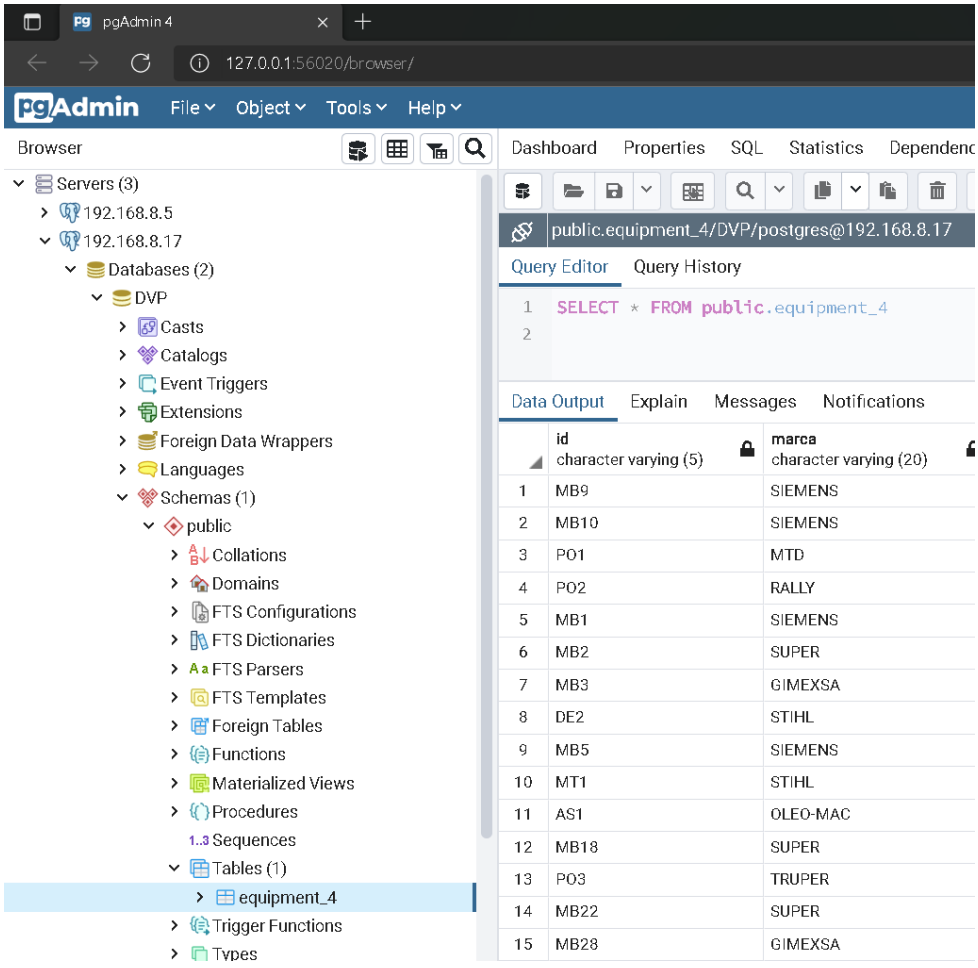


Figura 4.43 Fragmento del sitio 192.168.8.17

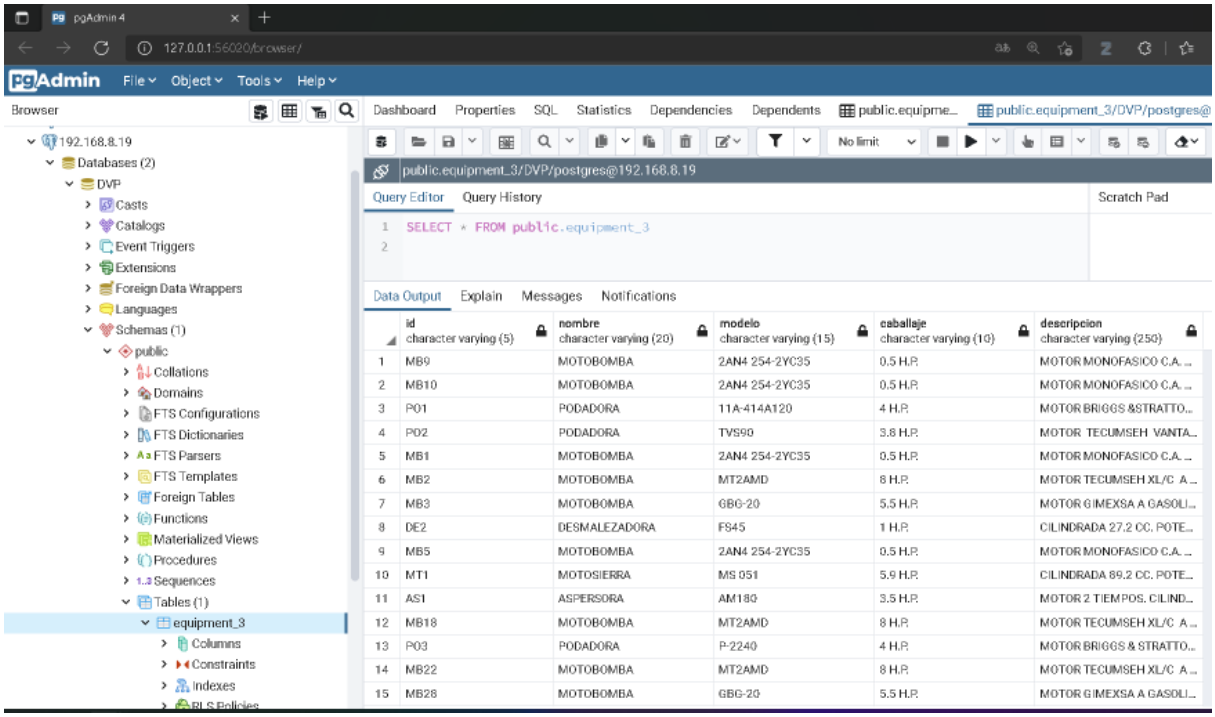


Figura 4.44 Fragmento del sitio 192.168.8.19

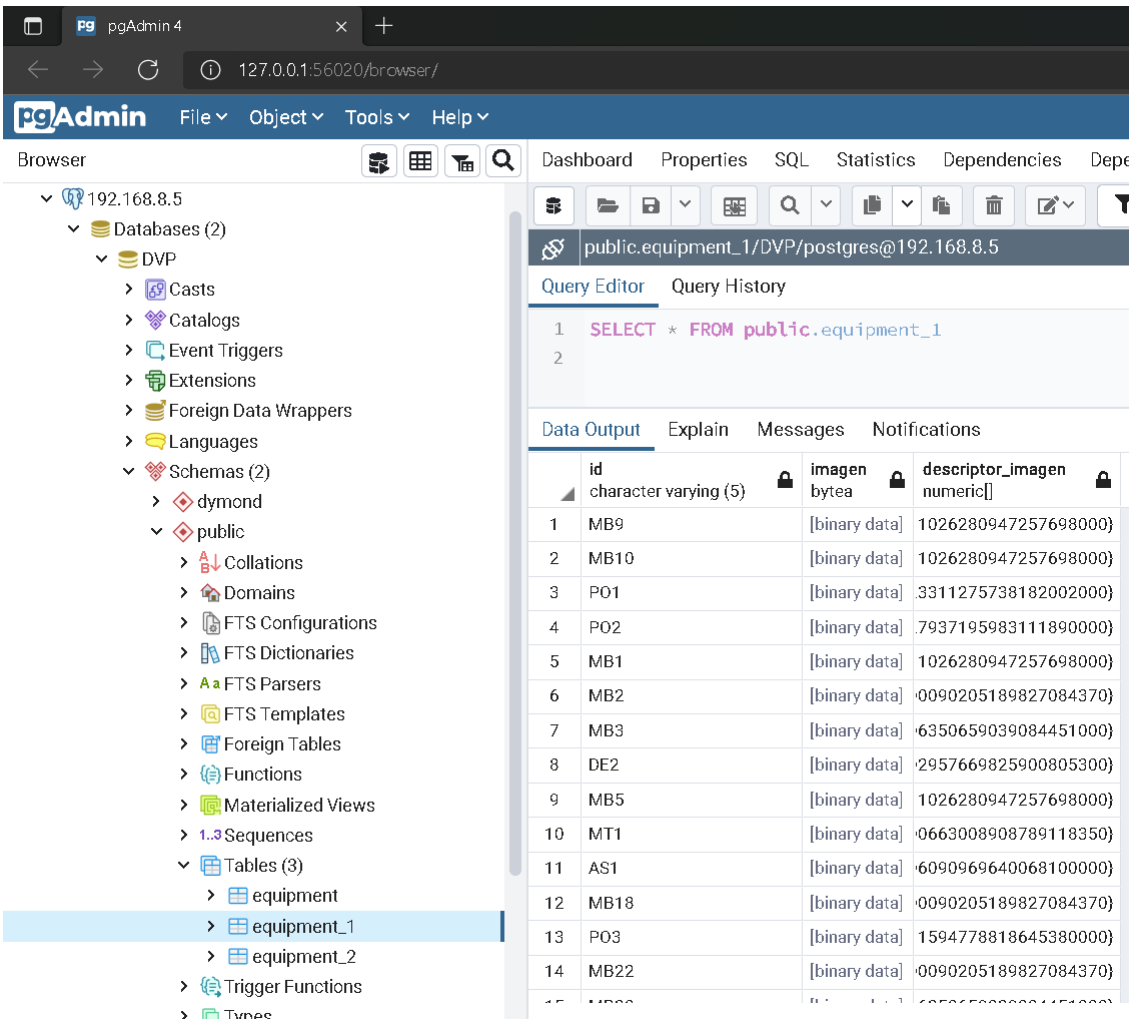


Figura 4.45 Sito 192.168.8.5 Fragmento multimedia *equipment_1*

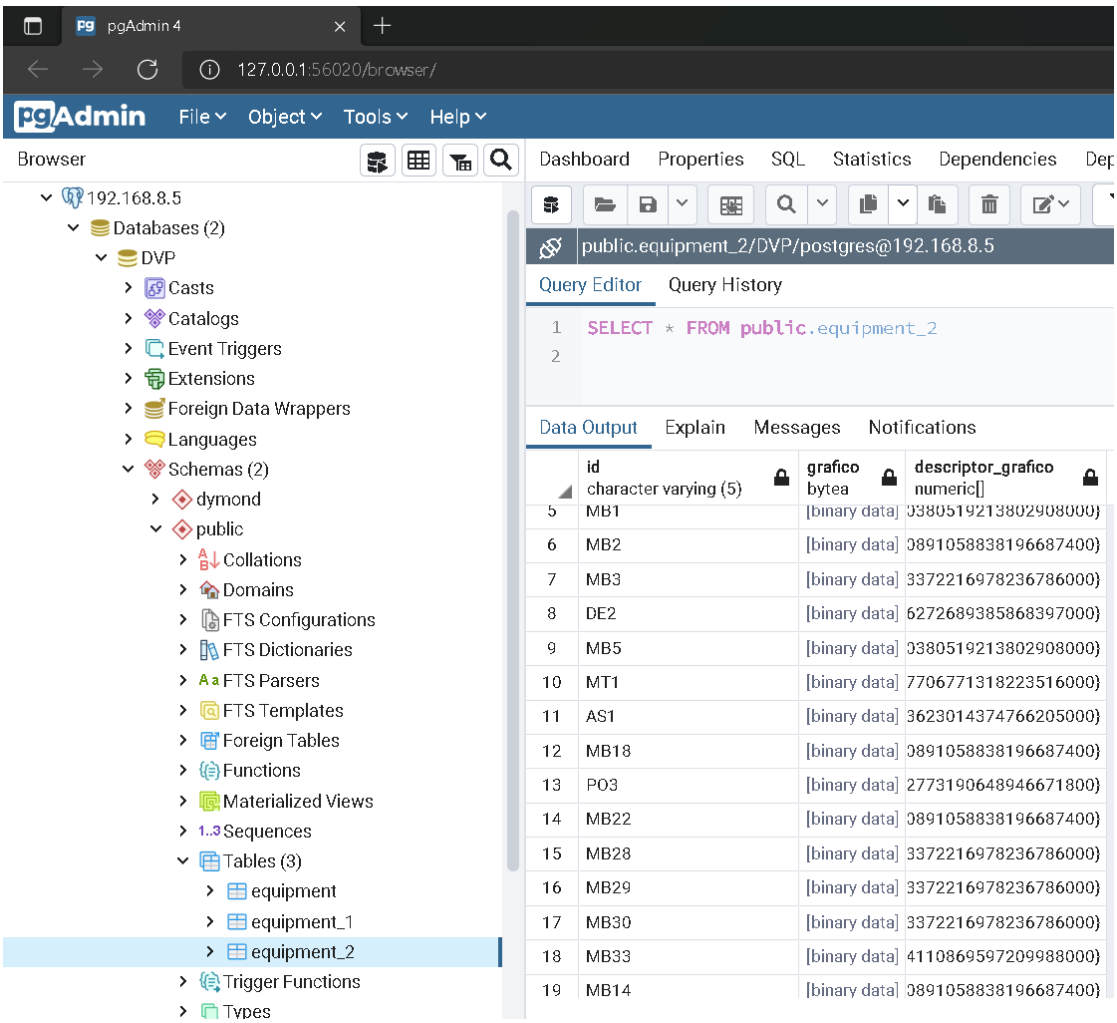


Figura 4.46 192.168.8.5 Fragmentos multimedia *equipment_2*

4.4 Evaluación del método desarrollado

El método de fragmentación desarrollado para la fragmentación vertical considerando consultas basadas en contenido (CBIRVF, por sus siglas en inglés) se comparó con el algoritmo MAVP (*Multimedia Adaptable Vertical Partitioning*, Fragmentación Vertical Adaptable Multimedia) [13]. Se utilizó el escenario de una base de datos multimedia simple usada para administrar equipo en una compañía de venta de maquinaria. La base de datos consiste de la tabla equipment (*id, nombre, marca, modelo, caballaje, descripción, imagen, descriptor_imagen, grafico, descriptor_grafico*) con 100 tuplas.

Se consideraron 3 sitios: 192.188.8.29, 192.188.8.33 y 192.188.8.5. La base de datos se encuentra en el tercer sitio. En cada sitio se ejecutaron las siguientes consultas:

Sitio 192.168.8.29

- q_1: DELETE FROM equipment WHERE marca=”MTD”
- q_2: UPDATE equipment SET nombre='MOTOBOMBA' WHERE marca! ="STIHL"
- q_3: SELECT marca,nombre FROM equipment WHERE modelo="MT2AMD”
- q_4: UPDATE equipment SET marca='SIEMENS' WHERE id="MB12"
- q_5: DELETE FROM equipment WHERE caballaje!=""
- q_6: SELECT nombre, descripcion FROM equipment WHERE id!="AS1"
- q_7: DELETE FROM equipment WHERE modelo="FS45"

Sitio 192.168.8.33

- q_1: UPDATE equipment SET nombre='MOTOBOMBA' WHERE marca="SUPER"
- q_2: DELETE FROM equipment WHERE modelo="FS45"
- q_3: SELECT nombre, marca, modelo FROM equipment WHERE id="Aldo"
- q_4: SELECT descripcion FROM equipment WHERE id!=""
- q_5: SELECT nombre, modelo FROM equipment WHERE marca!="MTD"
- q_6: SELECT marca FROM equipment WHERE id="MB12"
- q_7: SELECT descripcion FROM equipment WHERE marca!="SUPER"
- q_8: UPDATE equipment SET descripcion='MOTOR MONOFASICO' WHERE id="DE13"
- q_9: DELETE FROM equipment WHERE marca!="MTD"
- q_{10}: SELECT nombre, caballaje FROM equipment WHERE id="MB3"
- q_{11}: UPDATE equipment SET descripcion=' MOTOR' WHERE descripcion="CILINDRADA"
- q_{12}: UPDATE equipment SET caballaje='4 H.P.' WHERE id="PO4"

MAVP no considera el tipo de operación, no distingue entre consultas ejecutadas en distintos sitios y utiliza como entrada una matriz de uso de atributos (MUA) donde MUA (q_i, aj) =1 si la consulta q_i utiliza al atributo a_j y MUA (q_i, aj) =0, si no lo utiliza. F es la frecuencia de la consulta y T es el tamaño de los atributos en bytes. LA MUA obtenida por MAVP se muestra en la Tabla 4.1.

Tabla 4.1 Matriz de uso por atributo

Q/A	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	F
q_1	0	0	1	0	0	0	0	0	0	0	3
q_2	0	1	1	0	0	0	0	0	0	0	2
q_3	0	1	1	1	0	0	0	0	0	0	2
q_4	1	1	0	0	0	0	0	0	0	0	1
q_5	0	0	0	0	1	0	0	0	0	0	1
q_6	1	1	0	0	0	1	0	0	0	0	1
q_7	0	0	0	1	0	0	0	0	0	0	2
q_8	1	1	1	1	0	0	0	0	0	0	1
q_9	1	0	0	0	0	1	0	0	0	0	2
q_{10}	1	0	1	0	0	0	0	0	0	0	1
q_{11}	0	0	1	0	0	1	0	0	0	0	1
q_{12}	1	1	0	0	1	0	0	0	0	0	1
q_{13}	0	0	0	0	0	1	0	0	0	0	1
q_{14}	1	0	0	0	1	0	0	0	0	0	1
T	5	20	20	15	10	250	900	900	500	500	

a_1=*id*, a_2=*nombre*, a_3=*marca*, a_4=*modelo*, a_5=*caballaje*, a_6=*descripción*, a_7=*imagen*, a_8=*descriptor_imagen*, a_9=*grafico*, a_{10}=*descriptor_grafico*

El mejor esquema de fragmentación obtenido por MAVP es *equipment_1(id, nombre), equipment_2(id, marca), equipment_3(id, modelo), equipment_4(id, caballaje), equipment_5(id, descripción*), equipment_6(*id, imagen, descriptor_imagen, grafico, descriptor_grafico*). Todos los fragmentos se colocan en el tercer sitio porque MAVP no realiza la asignación de los fragmentos.

El mejor esquema de fragmentación obtenido por CBIRVF (método desarrollado) es equipment_1(*id, imagen, descriptor_imagen*), equipment_2(*id, grafico, descriptor_grafico*), equipment_3(*id, nombre, modelo, caballaje, descripción*), equipment_4(*id, marca*).

Se utilizó un modelo de costos que considera que el costo de una consulta se compone de su costo de acceso a atributos irrelevantes, su costo de transporte y su costo de reunión.

$$costo(q_i) = IAAC(q_i) + TC(q_i) + CR(q_i)$$

El costo de acceso a atributos irrelevantes se obtiene de la siguiente forma:

$$IAAC(q_i) = \sum_{acc(q_i,a_j)=0 \wedge a_j \in fr(q_i)} T_j \times F_i$$

Donde T_j es el tamaño de un atributo en bytes, F_i es la frecuencia de una consulta, $acc(q_i,a_j)=0$ se refiere a que la consulta q_i no utiliza al atributo a_j y $fr(q_i)$ es el fragmento accedido por q_i.

El costo de transporte se calcula de la siguiente manera:

$$TC(q_i) = \sum_{acc(q_i,a_j)=1 \wedge a_j \notin S(q_i)} T_j \times F_i^2$$

Donde $acc(q_i,a_j)=1$ se refiere a que la consulta q_i utiliza al atributo a_j y $S(q_i)$ es el sitio donde se realiza la consulta q_i.

El costo de reunión se obtiene con la siguiente ecuación:

$$CR(q_i) = NT * NR * F_i$$

Donde *NT* es el número de tuplas y *NR* es el número de reuniones.

El costo de las consultas basadas en contenido se incrementará considerablemente en el esquema obtenido por MAVP debido a que todos los atributos multimedia y descriptores están ubicados en un mismo fragmento, por lo que la cantidad de atributos irrelevantes accedidos por las consultas será muy alta. Considérense 2 consultas basadas en contenido que se ejecutan en el sitio 3. La

primera requiere acceder a los atributos *descriptor_imagen* e *imagen*, la segunda accede a *gráfico* y *descriptor_grafico*. En el siguiente gráfico se muestra que para algunas consultas tradicionales el método desarrollado supera al método MAVP, se muestra de color naranja el costo obtenido por CBIRVF y de color azul al resultado de MAVP, en las consultas basadas en contenido, el resultado es demasiado notable, mientras el costo para MAVP en este tipo de consultas es muy alto, para CBIRVF es un costo de 0.

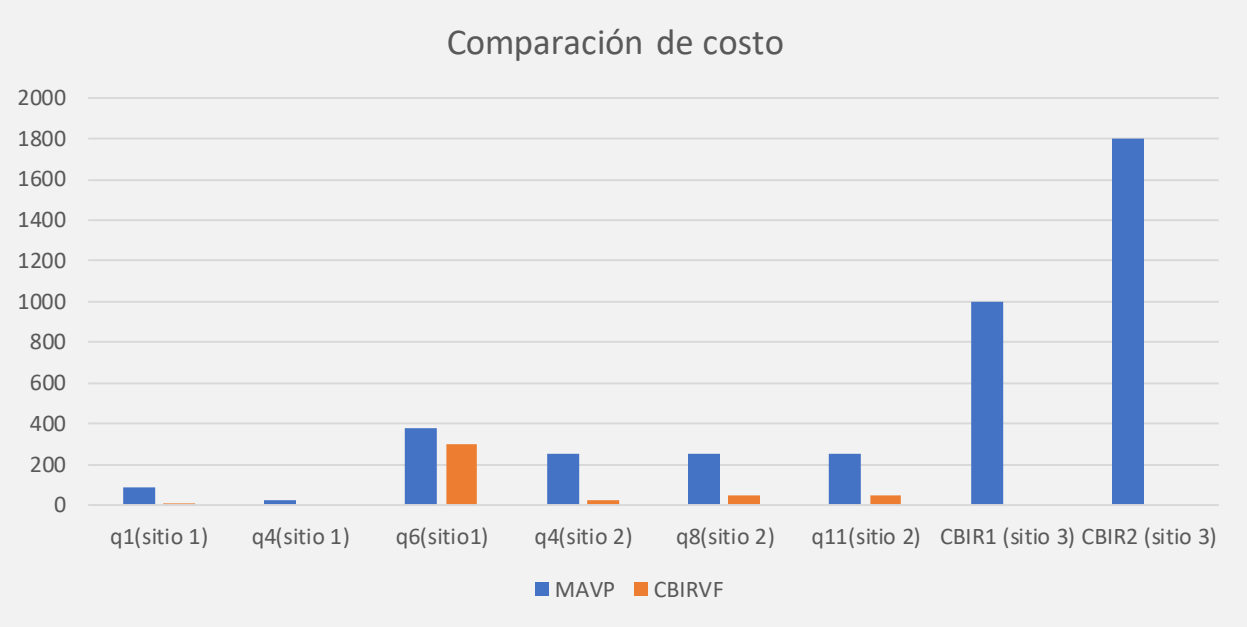


Figura 4.47 Comparación entre CBIRVF y MAVP

Capítulo 5. Conclusiones y Recomendaciones

En este capítulo se muestra la conclusión y se agrega una alternativa como recomendaciones en trabajos futuros.

5.1 Conclusiones

La fragmentación y asignación de datos en bases de datos son temas de gran interés es la industria, ya que permiten reducir el tiempo de respuesta y costo de ejecución de las consultas. Actualmente, existe un aumento exponencial en datos multimedia esto conlleva a la adaptación de los métodos de fragmentación en este tipo de bases de datos. Es por ello que este trabajo se centró en la fragmentación vertical para bases de datos multimedia. Las consultas basadas en contenido son necesarias para recuperar datos por medio de sus características visuales. Por tal motivo, el objetivo de este trabajo fue desarrollar un método de fragmentación vertical para base de datos multimedia que considere consultas basadas en contenido.

Gracias al análisis de las técnicas del estado del arte se demostró la carencia de trabajos de investigación que se enfoquen en fragmentación vertical de bases de datos multimedia y que consideren consultas basadas en contenido, por lo que en el análisis se eligió un método que fragmenta este tipo de bases de datos, pero no contempla consultas basadas en contenido (MAVP) para compararlo con la técnica desarrollada, llamada CBIRVF.

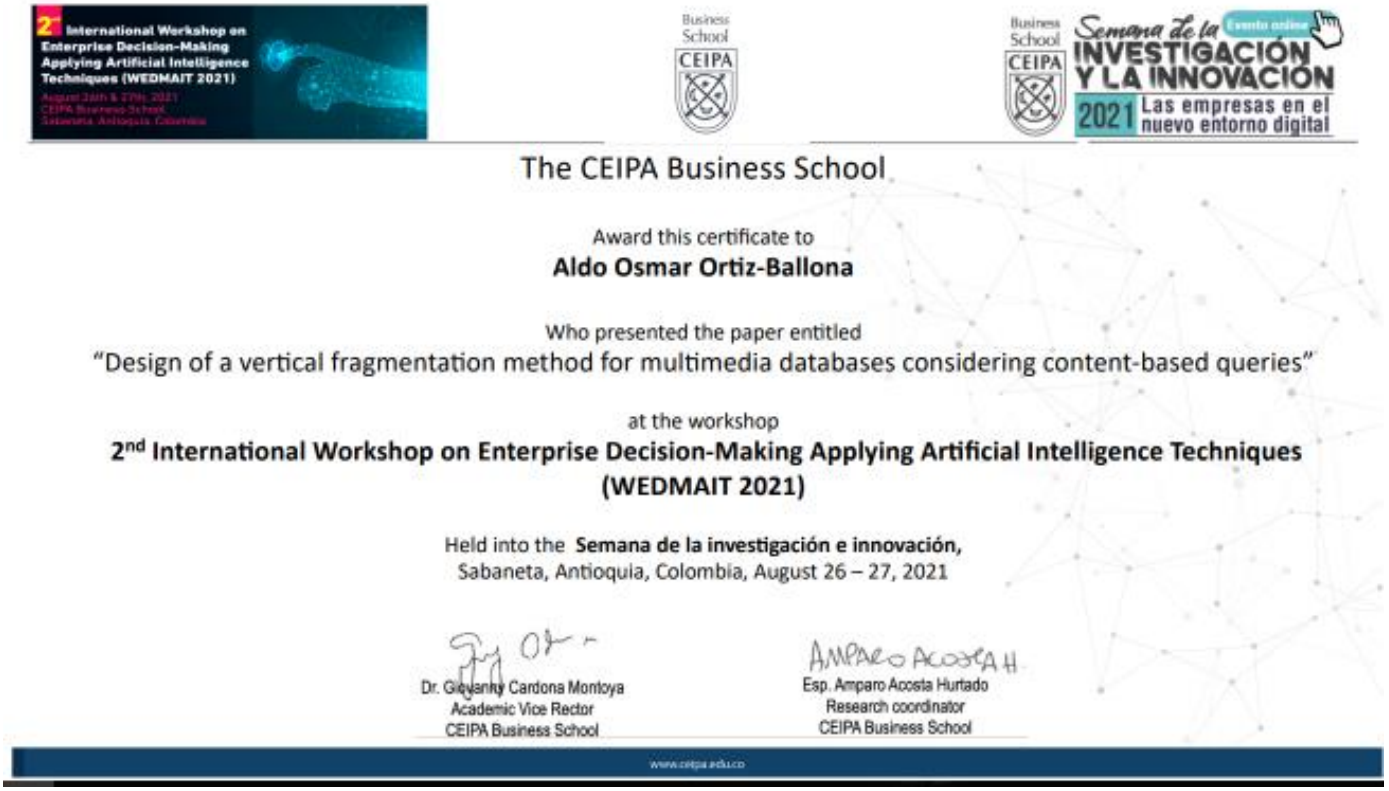
La comparación entre los dos enfoques reveló que CBIRVF logró una reducción considerable del costo de ejecución de consultas basadas en contenido utilizando un modelo de costos propuesto.

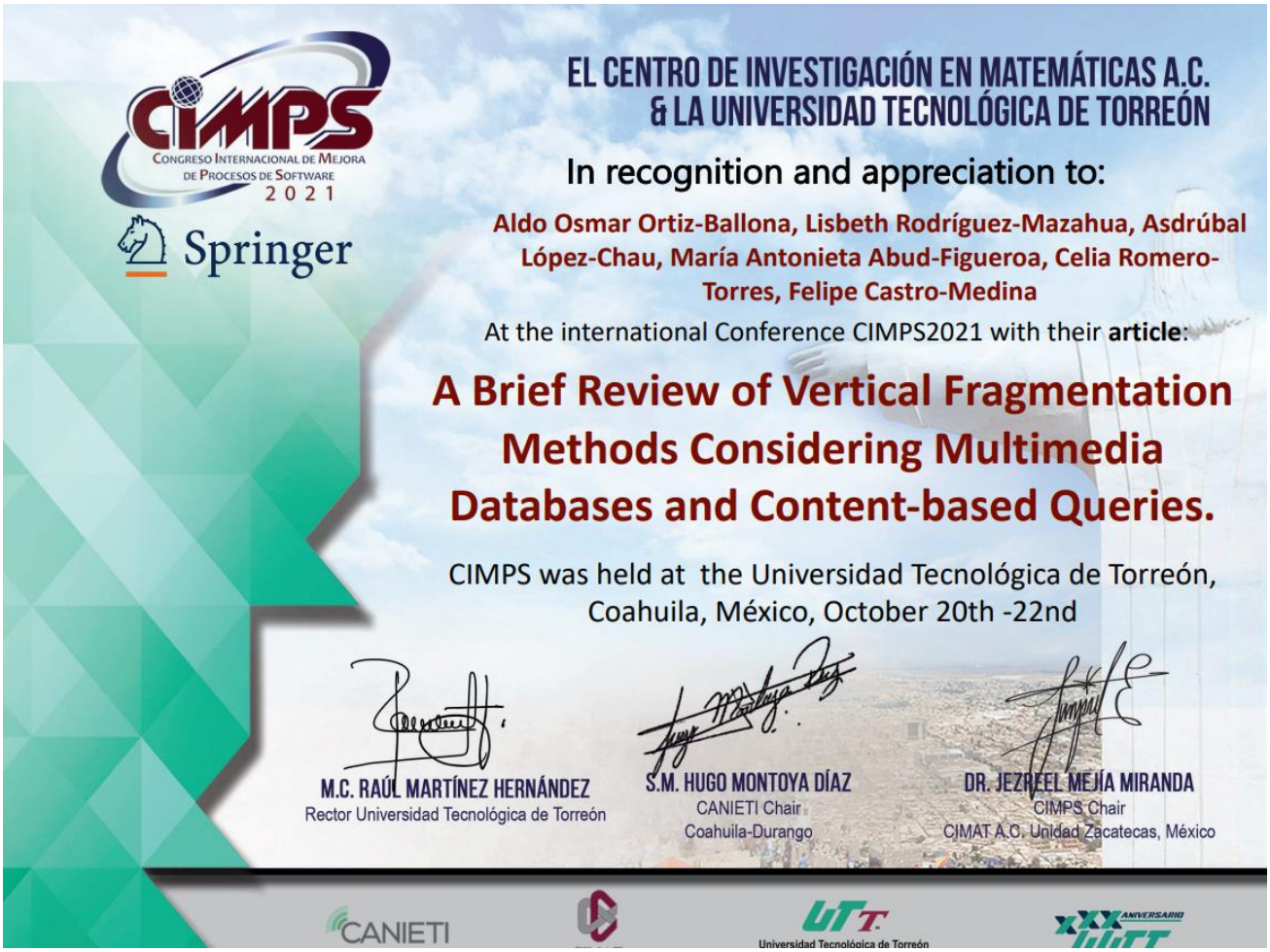
Con ese trabajo se benefician los investigadores y administradores de bases de datos que abordan los problemas de optimización de consultas en bases de datos multimedia, ya que se cuenta con un método capaz de proporcionar ventajas en el desempeño de consultas tradicionales y basadas en contenido.

5.2 Recomendaciones

La fragmentación estática acarrea problemas de mantenimiento, ya que el administrador de la base de datos debe calcular cuándo se debe de realizar una fragmentación nuevamente y esto es un problema para el DBA porque no sabría con certeza cuándo es el momento indicado para realizar esta tarea, es por esto que se propone como trabajo futuro agregar algún mecanismo que permita la fragmentación dinámica que esté basado en tiempos de ejecución por consulta realizada, es decir, que pasando cierto tiempo y cierto número de consultas realizadas, se analice el archivo de registro y que compare con el esquema anterior por si existen grandes cambios, se realice un nuevo esquema de fragmentación.

Productos académicos





Ortiz-Ballona A.O., Rodríguez-Mazahua L., López-Chau A., Abud-Figueroa M.A., Romero-Torres C., Castro-Medina F. (2022) **A Brief Review of Vertical Fragmentation Methods Considering Multimedia Databases and Content-Based Queries**. In: Mejia J., Muñoz M., Rocha Á., Avila-George H., Martínez-Aguilar G.M. (eds) New Perspectives in Software Engineering. CIMPS 2021. Advances in Intelligent Systems and Computing, vol 1416. Springer, Cham. https://doi.org/10.1007/978-3-030-89909-7_5

Referencias

[1] Z. Guo, Z. (Mark) Zhang, E. P. Xing, y C. Faloutsos, «Multimodal Data Mining in a Multimedia Database Based on Structured Max Margin Learning», *ACM Trans. Knowl. Discov. Data*, vol. 10, n.º 3, pp. 1-30, feb. 2016, doi: 10.1145/2742549.

[2] P. H. Oliveira *et al.*, «Employing Domain Indexes to Efficiently Query Medical Data From Multiple Repositories», *IEEE J. Biomed. Health Inform.*, vol. 23, n.º 6, pp. 2220-2229, nov. 2019, doi: 10.1109/JBHI.2018.2881381.

[3] M. T. Özsu y P. Valduriez, *Principles of Distributed Database Systems*. Cham: Springer International Publishing, 2020. doi: 10.1007/978-3-030-26253-2.

[4] R. Chbeir y D. Laurent, «Towards a novel approach to multimedia data mixed fragmentation», en *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*, New York, NY, USA, oct. 2009, pp. 200-204. doi: 10.1145/1643823.1643860.

[5] S. Saad, J. Tekli, R. Chbeir, y K. Yetongnon, «Towards Multimedia Fragmentation», en *Advances in Databases and Information Systems*, Berlin, Heidelberg, 2006, pp. 415-429. doi: 10.1007/11827252_31.

[6] F. Getahun, J. Tekli, S. Atnafu, y R. Chbeir, «The use of semantic-based predicates implication to improve horizontal multimedia database fragmentation», en *Workshop on multimedia information retrieval on The many faces of multimedia semantics*, New York, NY, USA, sep. 2007, pp. 29-38. doi: 10.1145/1290067.1290073.

[7] F. Getahun, J. Tekli, S. Atnafu, y R. Chbeir, «Towards efficient horizontal multimedia database fragmentation using semantic-based predicates implication», *Sbbd*, pp. 68-82.

[8] L. Rodríguez-Mazahua, G. Alor-Hernández, Ma. A. Abud-Figueroa, y S. G. Peláez-Camarena, «Horizontal Partitioning of Multimedia Databases Using Hierarchical Agglomerative

Clustering», en *Nature-Inspired Computation and Machine Learning*, Cham, 2014, pp. 296-309. doi: 10.1007/978-3-319-13650-9_27.

[9] C. Fung, K. Karlapalem, y Q. Li, «An evaluation of vertical class partitioning for query processing in object-oriented databases», *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, n.º 5, pp. 1095-1118, sep. 2002, doi: 10.1109/TKDE.2002.1033777.

[10] C.-W. Fung, K. Karlapalem, y Q. Li, «Cost-driven vertical class partitioning for methods in object oriented databases», *VLDB*, vol. 12, n.º 3, pp. 187-210, oct. 2003, doi: 10.1007/s00778-002-0084-7.

[11] N. Thakur y B. Ram, «Examining the Performance of Vertical Fragmentation using FP-MAX Algorithm», *IJCA*, vol. 116, n.º No. 23, abr. 2015.

[12] C. Fung, E. W. Leung, y Q. Li, «Efficient Query Execution Techniques in a 4DIS Video Database System for eLearning», *Multimedia Tools and Applications*, vol. 20, n.º 1, pp. 25-49, may 2003, doi: 10.1023/A:1023418316038.

[13] L. Rodriguez y X. Li, «A Vertical Partitioning Algorithm for Distributed Multimedia Databases», en *Database and Expert Systems Applications*, vol. 6861, A. Hameurlain, S. W. Liddle, K.-D. Schewe, y X. Zhou, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 544-558. doi: 10.1007/978-3-642-23091-2_48.

[14] L. Rodríguez, X. Li, J. Cervantes, y F. García-Lamont, «DYMOND: an active system for dynamic vertical partitioning of multimedia databases», en *Proceedings of the 16th International Database Engineering & Applications Sysmposium*, New York, NY, USA, ago. 2012, pp. 71-80. doi: 10.1145/2351476.2351485.

[15] L. Rodríguez, X. Li, A. D. Cuevas-Rasgado, y F. García-Lamont, «DYVEP: An active database system with vertical partitioning functionality», en *2013 10th IEEE INTERNATIONAL CONFERENCE ON NETWORKING, SENSING AND CONTROL (ICNSC)*, abr. 2013, pp. 457-462. doi: 10.1109/ICNSC.2013.6548782.

[16] L. Rodríguez-Mazahua, G. Alor-Hernández, X. Li, J. Cervantes, y A. López-Chau, «Active rule base development for dynamic vertical partitioning of multimedia databases», *J Intell Inf Syst*, vol. 48, n.º 2, pp. 421-451, abr. 2017, doi: 10.1007/s10844-016-0420-9.

[17] M. S. Lew, N. Sebe, C. Djeraba, y R. Jain, «Content-based multimedia information retrieval: State of the art and challenges», *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 2, n.º 1, pp. 1-19, feb. 2006, doi: 10.1145/1126004.1126005.

[18] S. Mehta, P. Agarwal, P. Shrivastava, y J. Barlawala, «Differential bond energy algorithm for optimal vertical fragmentation of distributed databases», *Journal of King Saud University - Computer and Information Sciences*, p. S1319157818302519, sep. 2018, doi: 10.1016/j.jksuci.2018.09.020.

[19] H. Rahimi, F.-A. Parand, y D. Riahi, «Hierarchical simultaneous vertical fragmentation and allocation using modified Bond Energy Algorithm in distributed databases», *Applied Computing and Informatics*, vol. 14, n.º 2, pp. 127-133, jul. 2018, doi: 10.1016/j.aci.2015.03.001.

[20] V. N. Luong, V. K. Solanki, y N. H. H. Cuong, «Fragmentation in Distributed Database Design Based on Ant Colony Optimization Technique», *IJIRR*, vol. 9, n.º 2, pp. 28-37, abr. 2019, doi: 10.4018/IJIRR.2019040103.

[21] L. Rodríguez y X. Li, «Dynamic Vertical Partitioning of Multimedia Databases Using Active Rules», en *Database and Expert Systems Applications*, Berlin, Heidelberg, 2012, pp. 191-198. doi: 10.1007/978-3-642-32597-7_17.

[22] X. Gu, X. Yang, W. Wang, Y. Jin, y D. Meng, «CHAC: An Effective Attribute Clustering Algorithm for Large-Scale Data Processing», en *2012 IEEE Seventh International Conference on Networking, Architecture, and Storage*, Xiamen, China, jun. 2012, pp. 94-98. doi: 10.1109/NAS.2012.16.

[23] W. Zhao, Y. Cheng, y F. Rusu, «Vertical partitioning for query processing over raw data», en *Proceedings of the 27th International Conference on Scientific and Statistical Database Management*, La Jolla California, jun. 2015, pp. 1-12. doi: 10.1145/2791347.2791369.

[24] G. Campero Durand, R. Piriyev, M. Pinnecke, D. Broneske, B. Gurumurthy, y G. Saake, «Automated Vertical Partitioning with Deep Reinforcement Learning», en *New Trends in Databases and Information Systems*, vol. 1064, T. Welzer, J. Eder, V. Podgorelec, R. Wrembel, M. Ivanović, J. Gamper, M. Morzy, T. Tzouramanis, J. Darmont, y A. Kamišalić Latifić, Eds. Cham: Springer International Publishing, 2019, pp. 126-134. doi: 10.1007/978-3-030-30278-8_16.

[25] E. Costa, C. Costa, y M. Y. Santos, «Evaluating partitioning and bucketing strategies for Hive-based Big Data Warehousing systems», *J Big Data*, vol. 6, n.º 1, p. 34, dic. 2019, doi: 10.1186/s40537-019-0196-1.

[26] A. A. Amer, «On K-means clustering-based approach for DDBSs design», *Journal of Big Data*, vol. 7, n.º 1, p. 31, may 2020, doi: 10.1186/s40537-020-00306-9.

[27] A. E. A. Raouf, N. L. Badr, y M. F. Tolba, «An Enhanced CRUD for Vertical Fragmentation Allocation and Replication Over the Cloud Environment», en *Proceedings of the 10th International Conference on Informatics and Systems - INFOS '16*, Giza, Egypt, 2016, pp. 146-152. doi: 10.1145/2908446.2908480.

[28] A. A. Amer, M. H. Mohamed, y K. Al_Asri, «ASGOP: An aggregated similarity-based greedy-oriented approach for relational DDBSs design», *Heliyon*, vol. 6, n.º 1, p. e03172, ene. 2020, doi: 10.1016/j.heliyon.2020.e03172.

[29] A. Dahal y S. R. Joshi, «A Clustering Based Vertical Fragmentation and Allocation of a Distributed Database», en *2019 Artificial Intelligence for Transforming Business and Society (AITB)*, Kathmandu, Nepal, nov. 2019, pp. 1-5. doi: 10.1109/AITB48515.2019.8947444.

[30] R. Rojas Ruiz, L. Rodríguez-Mazahua, A. López-Chau, S. G. Peláez-Camarena, M. A. Abud-Figueroa, y I. Machorro-Cano, «A CBIR System for the Recognition of Agricultural Machinery», *RCS*, vol. 147, n.º 3, pp. 9-16, dic. 2018, doi: 10.13053/rcs-147-3-1.

[31] S. Bhardwaj, G. Pandove, y P. K. Dahiya, «A Futuristic Hybrid Image Retrieval System based on an Effective Indexing Approach for Swift Image Retrieval», 2020.

https://www.semanticscholar.org/paper/A-Futuristic-Hybrid-Image-Retrieval-System-based-on-Bhardwaj-Pandove/6153c18cfb29817aef6dff082fe4c86942c98cdf (accedido nov. 04, 2021).

[32] A. A. Amer, M. H. Mohamed, A. A. Sewisy, y K. Al Asri, «An Aggregated Similarity Based Hierarchical Clustering Technique for Relational DDBS Design», en *2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, Solan Himachal Pradesh, India, dic. 2018, pp. 295-299. doi: 10.1109/PDGC.2018.8745981.

[33] L. Rodriguez y X. Li, «A support-based vertical partitioning method for database design», en *2011 8th International Conference on Electrical Engineering, Computing Science and Automatic Control*, Merida City, Mexico, oct. 2011, pp. 1-6. doi: 10.1109/ICEEE.2011.6106682.

[34] M. J. Rodríguez-Arauz, L. Rodríguez-Mazahua, M. L. Arrioja-Rodríguez, A. Abud-Figueroa, y S. G. Peláez-Camarena, «Design of a Multimedia Data Management System that Uses Horizontal Fragmentation to Optimize Content-based Queries», p. 7, 2020.

[35] «Conozca más sobre la tecnología Java». https://www.java.com/es/about/ (accedido nov. 28, 2021).

[36] «JavaServer Faces Technology». https://www.oracle.com/java/technologies/javaserverfaces.html (accedido nov. 28, 2021).

[37] «Welcome to Apache NetBeans». https://netbeans.apache.org/ (accedido nov. 28, 2021).

[38] C. Nieves Guerrero, J. Ucán Pech, y V. Menéndez Domínguez, «UWE en Sistema de Recomendación de Objetos de Aprendizaje. Aplicando Ingeniería Web: Un Método en Caso de Estudio», *REVISTA LATINOAMERICANA DE INGENIERIA DE SOFTWARE*, vol. 2, pp. 137-143, jun. 2014, doi: 10.18294/relais.2014.137-143.

[39] «Overview | Postgres-XL». https://www.postgres-xl.org/overview/ (accedido nov. 28, 2021).

[40] A. A. Amer, M. H. Mohamed, y K. Al_Asri, «On an Effective Hierarchical Clustering Based Model for Data Fragmentation and Allocation in Relational DDBS: Review and Proposal»,

en *Proceedings of the 4th ACM International Conference of Computing for Engineering and Sciences on - ICCES'18*, Kuala Lumpur, Malaysia, 2018, pp. 1-9. doi: 10.1145/3213187.3293604.

[41] S. Zhang y K. Zhao, «A New Method for Computation of the Node's Estimated_Cost of Transaction_Based on Approach in Vertical Partitioning», en *2011 International Conference on Internet Technology and Applications*, Wuhan, China, ago. 2011, pp. 1-3. doi: 10.1109/ITAP.2011.6006251.

[42] A. A. Amer y H. I. Abdalla, «An integrated design scheme for performance optimization in distributed environments», en *International Conference on Education and e-Learning Innovations*, Sousse, jul. 2012, pp. 1-8. doi: 10.1109/ICEELI.2012.6360610.

[43] Z. Chen, S. Yang, H. Zhao, y H. Yin, «An Objective Function for Dividing Class Family in NoSQL Database», en *2012 International Conference on Computer Science and Service System*, Nanjing, China, ago. 2012, pp. 2091-2094. doi: 10.1109/CSSS.2012.520.

[44] A. E. Abdel Raouf, N. L. Badr, y M. F. Tolba, «An optimized scheme for vertical fragmentation, allocation and replication of a distributed database», en *2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*, Cairo, dic. 2015, pp. 506-513. doi: 10.1109/IntelCIS.2015.7397268.

[45] L.-Y. Ho, M.-J. Hsieh, J.-J. Wu, y P. Liu, «Data Partition Optimization for Column-Family NoSQL Databases», en *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, Chengdu, China, dic. 2015, pp. 668-675. doi: 10.1109/SmartCity.2015.146.

[46] D. Kim, M. Kim, K. Kim, M. Sung, y W. W. Ro, «Dynamic Load Balancing of Parallel SURF with Vertical Partitioning», *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, n.º 12, pp. 3358-3370, dic. 2015, doi: 10.1109/TPDS.2014.2372763.

[47] L. Birhanu, S. Atnafu, y F. Getahun, «Native XML Document Fragmentation Model», en *2010 Sixth International Conference on Signal-Image Technology and Internet Based Systems*, Kuala Lumpur, dic. 2010, pp. 233-240. doi: 10.1109/SITIS.2010.47.

[48] S. Jagannatha, S. V. P. Reddy, T. V. S. Kumar, y K. R. Kanth, «Simulation and analysis of performance prediction in Distributed Database design using OO approach», en *2013 3rd IEEE International Advance Computing Conference (IACC)*, Ghaziabad, feb. 2013, pp. 1324-1329. doi: 10.1109/IAdCC.2013.6514420.

[49] R. R. Amossen, «Vertical partitioning of relational OLTP databases using integer programming», en *2010 IEEE 26th International Conference on Data Engineering Workshops (ICDEW 2010)*, Long Beach, CA, USA, 2010, pp. 93-98. doi: 10.1109/ICDEW.2010.5452739.

[50] Y.-F. Huang y C.-J. Lai, «Integrating frequent pattern clustering and branch-and-bound approaches for data partitioning», *Information Sciences*, vol. 328, pp. 288-301, ene. 2016, doi: 10.1016/j.ins.2015.08.047.

[51] R. A. Pazos, G. Vázquez, J. A. Martínez, J. Pérez-Ortega, y G. Martínez-Luna, «Minimizing roundtrip response time in distributed databases with vertical fragmentation», *Journal of Computational and Applied Mathematics*, p. 9, 2014.

[52] T. Tsuchida, T. Tsuji, y K. Higuchi, «Implementing Vertical Splitting for Large Scale Multidimensional Datasets and Its Evaluations», en *Data Warehousing and Knowledge Discovery*, Berlin, Heidelberg, 2011, pp. 208-223. doi: 10.1007/978-3-642-23544-3_16.

[53] N. Bobrov, G. Chernishev, y B. Novikov, «Workload-Independent Data-Driven Vertical Partitioning», en *New Trends in Databases and Information Systems*, Cham, 2017, pp. 275-284. doi: 10.1007/978-3-319-67162-8_27.

[54] K. Kaur y V. Laxmi, «A Novel Method of Data Partitioning Using Genetic Algorithm Work Load Driven Approach Utilizing Machine Learning», en *Cognitive Computing in Human Cognition: Perspectives and Applications*, P. K. Mallick, P. K. Pattnaik, A. R. Panda, y V. E. Balas, Eds. Cham: Springer International Publishing, 2020, pp. 49-60. doi: 10.1007/978-3-030-48118-6_5.

[55] M. Goli y S. M. T. Rouhani Rankoohi, «A new vertical fragmentation algorithm based on ant collective behavior in distributed database systems», *Knowl Inf Syst*, vol. 30, n.º 2, pp. 435-455, feb. 2012, doi: 10.1007/s10115-011-0384-6.

[56] R. Dharavath, V. Kumar, C. Kumar, y A. Kumar, «An Apriori-Based Vertical Fragmentation Technique for Heterogeneous Distributed Database Transactions», en *Intelligent Computing, Networking, and Informatics*, vol. 243, D. P. Mohapatra y S. Patnaik, Eds. New Delhi: Springer India, 2014, pp. 687-695. doi: 10.1007/978-81-322-1665-0_69.

[57] A. Ghorbanian, Y. Maghsoudi, y A. Mohammadzadeh, «Clustering-Based Band Selection Using Structural Similarity Index and Entropy for Hyperspectral Image Classification», *TS*, vol. 37, n.º 5, pp. 785-791, nov. 2020, doi: 10.18280/ts.370510.

[58] D. Kishore y C. Rao, «A Multi-class SVM Based Content Based Image Retrieval System Using Hybrid Optimization Techniques», *TS*, vol. 37, n.º 2, pp. 217-226, abr. 2020, doi: 10.18280/ts.370207.

[59] M. Buvana, K. Muthumayil, y T. Jayasankar, «Content-Based Image Retrieval based on Hybrid Feature Extraction and Feature Selection Technique Pigeon Inspired based Optimization», vol. 25, n.º 1, p. 20, 2021.

[60] H. Abdalla y A. M. Artoli, «Towards an Efficient Data Fragmentation, Allocation, and Clustering Approach in a Distributed Environment», *Information*, vol. 10, n.º 3, Art. n.º 3, mar. 2019, doi: 10.3390/info10030112.