Pro Sciences
Revista de Producción, Ciencias e Investigación

Uso de grafos para el análisis de datos del sector de ventas: un caso de estudio

Using graphs for data analysis in the sales sector: a case study

Nuvia Beltrán Robayo
nbeltran@uagraria.edu.ec
Universidad Agraria del Ecuador, Ecuador

Andrés Medina Robayo
aimedina@uagraria.edu.ec
Universidad Agraria del Ecuador, Ecuador

Darwin Pow Chon Long Vásquez
dpow@uagraria.edu.ec
Universidad Agraria del Ecuador, Ecuador

Johanna Sánchez Guerrero
jsanchez@uagraria.edu.ec
Universidad Agraria del Ecuador, Ecuador

Pro Sciences: Revista de Producción, Ciencias e Investigación
CIDEPRO, Ecuador
e-ISSN: 2588-1000
Periodicidad: Trimestral
Vol. 8, No. 51, 2024
editor@journalprosciences.com

Recepción: 10 enero 2024
Aprobación: 25 febrero2024

DOI:https://doi.org/10.29018/issn.2588-1000vol7iss51.2024pp37-58

Resumen: El gran volumen y la estructura de datos que se maneja en la actualidad ha creado la necesidad de utilizar bases de datos no convencionales denominada Nosql que por sus características permiten la manipulación de los datos de forma más rápida, entre ellas se encuentra una orientada a grafos entendiéndose a este como un esquema que maneja nodos y relaciones existentes entre entidades. Por lo cual en el presente trabajo se realizó un análisis de datos del área de venta de una plataforma dedicada al e-commerce denominada Flipkart la cual se obtuvo de Kaggle, el registro constaba de 17 campos con los cuales se procedió a la creación de la base de datos en Neo4j, además de esquemas y aplicación del lenguaje Cypher para las consultas visualizando patrones de ventas sobre categorías y los descuentos aplicados siendo el de mayor uso el del 60%, todos estos datos pueden ser útiles para la toma de decisiones informada y generación de estrategias según la ubicación de los clientes así como el reconocimiento de categorías de productos con baja calificación. En conclusión, este tipo de base de datos permite ver los datos de forma gráfica definiendo las relaciones existentes, permitiendo a los usuarios visualizar de forma rápida y sencilla patrones que tienen los datos, sin embargo, es recomendable que para mejores resultados se aplique un proceso de extracción, transformación y carga (ETL), a su vez verificar las relaciones que existen entre las diferentes entidades para una mejor visualización.

Palabras clave: análisis, grafo, neo4j, NoSQL, ventas

Pro Sciences
Revista de Producción, Ciencias e Investigación

Como citar: Beltrán Robayo, N., Medina Robayo, A., Long Vásquez, D. P. C., & Sánchez Guerrero, J. (2024). Uso de grafos para el análisis de datos del sector de ventas: un caso de estudio. *Pro Sciences: Revista De Producción, Ciencias E Investigación*, *8*(51). Recuperado a partir de https://journalprosciences.com/index.php/ps/article/view/688

Abstract The large volume and structure of data that is currently handled has created the need to use unconventional databases called Nosql, which due to their characteristics allow the manipulation of data more quickly, among them is one oriented to graphs. understanding this as a scheme that manages nodes and existing relationships between entities. Therefore, in this work, a data analysis of the sales area of a platform dedicated to e-commerce called Flipkart was carried out, which was obtained from Kaggle, the record consisted of 17 fields with which the creation of the database in Neo4j, in addition to schemas and application of the Cypher language for queries visualizing sales patterns on categories and the discounts applied, the most used being 60%, all this data can be useful for informed decision making and generation of strategies according to the location of customers as well as the recognition of product categories with low ratings. In conclusion, this type of database allows you to view the data graphically, defining the existing relationships, allowing users to quickly and easily visualize patterns in the data. However, it is recommended that a process be applied for better results. extraction, transformation and loading (ETL), in turn verifying the relationships that exist between the different entities for a better visualization.

Keywords:*Analysis, Graph, Neo4j, NoSQL, Sales*

INTRODUCCIÓN

En la actualidad se está manejando gran cantidad de información y procesarla de forma rápida en ocasiones toma demasiado tiempo, es por ello que algunas empresas están optando por el uso de base de datos Nosql, ya que las relacionales tienen ciertas restricciones al momento de manejar los datos; en este contexto uno de las bases de datos de esta

característica son las basadas en grafos las cuales permiten realizar consultas y que sus resultados se presente de forma gráfica identificando relaciones entre los registros.

En el área de ventas de cualquier empresa es imprescindible contar con herramientas que permitan verificar los productos más vendidos o si alguna estrategia comercial está generando utilidad, por ende algunas utilizan plataformas para el análisis de datos que van desde un simple Excel hasta el uso de herramientas analíticas, otras generan informes personalizados a través de aplicaciones lo que implica realizar en muchas ocasiones consultas complejas con sus datos y más si se lleva sus registros en una base de datos relacional, sin embargo, estas en muchas ocasiones no muestran los datos de forma intuitiva y simple comprensión, es por ello que en el presente trabajo se analizó los datos del área de ventas relacionadas a la plataforma de e-commerce Flipkart para verificar cuáles han sido las categorías y productos más cotizados, las calificaciones que obtuvieron los productos, las tiendas que vendieron más así como las ciudades.

Por ende, el objetivo del trabajo fue analizar los datos de ventas relacionadas a la plataforma de e-commerce Flikpkart utilizando Neo4j para la identificación de las tendencias que ha tenido la palataforma, mientras que los objetivos específicos fueron recopilar datos de una empresa dedicada a las ventas para la subida de los registros a Neo4j, también construir los esquemas mediante la aplicación de consultas para el establecimiento de tendencias de ventas y por último realizar la interpretación de los datos identificando los patrones de compra y relaciones de diferentes campos.

Cabe indicar que el uso de grafos en el área de ventas puede ser de gran beneficio en las empresas ya que permiten identificar de forma rápida las relaciones entre los datos y patrones que se tiene porque cada consulta realizada se esquematiza de forma gráfica.

Grafo

La historia de la teoría de grafos se remonta al siglo XVIII, cuando el matemático alemán Leonhard Euler resolvió el problema de los puentes de Königsberg. La teoría de grafos es una rama de la matemática que estudia las relaciones entre objetos, utilizando estructuras matemáticas llamadas grafos. Los grafos son estructuras matemáticas que representan relaciones entre objetos.

Un grafo está compuesto por nodos, que representan los objetos, y aristas, que representan las relaciones entre los objetos. Los grafos se pueden utilizar para representar una variedad de relaciones, como las relaciones sociales, las relaciones comerciales, las relaciones genéticas, las relaciones espaciales, etc (Zaki y Meira, 2014).

Un grafo se puede representar de forma gráfica, mediante un diagrama en el que los nodos se representan como puntos y las aristas se representan como líneas que conectan a los nodos.
El trabajo de Euler fue el primer paso en el desarrollo de la teoría de grafos. En el siglo XIX, otros matemáticos, como Carl Friedrich Gauss y Arthur Cayley, continuaron desarrollando el campo. En el siglo XX, la teoría de grafos se convirtió en una disciplina matemática formal, con numerosas aplicaciones en diversas áreas, como las matemáticas, la informática, la ingeniería y las ciencias sociales (Arenado, 2023).

En la actualidad los grafos de datos son una herramienta poderosa para el análisis de datos ya que pueden integrar información de diferentes fuentes, lo que les permite descubrir patrones y relaciones que no serían visibles de otro modo (Saorín, 2019).

Base de Datos NoSQL

Una base de datos no relacional, también conocida como base de datos NoSQL, es un sistema de gestión de bases de datos que no se basa en el modelo relacional. Las bases de datos relacionales almacenan datos en tablas que se relacionan entre sí mediante claves. En cambio, las bases de datos no SQL utilizan otros modelos, como el modelo de grafos, el modelo de documentos o el modelo de clave-valor (Wu, 2021).

Las bases de datos no SQL se utilizan para almacenar datos que no se ajustan bien al modelo relacional, como datos de redes sociales, datos de sensores o datos de geolocalización; se utilizan a menudo para aplicaciones que requieren un alto rendimiento, escalabilidad o flexibilidad.

Características

Las bases de datos NoSQL son un conjunto de sistemas de gestión de bases de datos que se distinguen de las bases de datos relacionales tradicionales en varios aspectos. Este tipo de base de

datos no requieren un lenguaje de consulta estructurado (SQL) como lenguaje principal, suelen tener una interfaz más simple que es más fácil de aprender y usar. Por otra

parte, no requieren una estructura fija y tabular (Migani et al., 2018). Las bases de datos relacionales requieren que los datos se organicen en tablas, con filas y columnas.

Las bases de datos NoSQL, por otro lado, pueden almacenar datos de una variedad de formas, como listas, árboles, grafos y documentos, esto les permite adaptarse mejor a una amplia gama de aplicaciones.

Además, no soportan operaciones JOIN, las cuales son utilizadas en base de datos relacionales para consultar varias tablas, logrando ser más eficientes para operaciones simples, como la búsqueda y la actualización de datos. Entre otras de las características se tiene que las bases de datos NoSQL no garantizan por completo las propiedades de ACID (atomicidad, consistencia, aislamiento y durabilidad).

Las propiedades de ACID garantizan que las transacciones se ejecuten de forma consistente y atómica, incluso en caso de fallas. Las bases de datos NoSQL, por otro lado, pueden comprometer la consistencia para mejorar el rendimiento y la escalabilidad. A su vez, se basan en la escalabilidad horizontal, esto significa que pueden ampliarse agregando más servidores. Las bases de datos relacionales, por otro lado, suelen ser más difíciles de escalar horizontalmente (Marrero et al., 2023).

En conclusión, las bases de datos NoSQL ofrecen una variedad de ventajas sobre las bases de datos relacionales tradicionales. Son más fáciles de aprender y usar, pueden adaptarse mejor a una amplia gama de aplicaciones, son más eficientes para operaciones simples y se pueden escalar más fácilmente.

Categoría de almacenamiento en NoSQL

Las bases de datos NoSQL se caracterizan por su flexibilidad y escalabilidad, lo que las hace adecuadas para una variedad de aplicaciones. Existen cuatro tipos principales de bases de datos NoSQL:

Almacenamiento clave-valor

Se caracteriza porque es el más simple y eficiente. Los datos se almacenan como pares de clave y valor, donde la clave es un identificador único y el valor puede ser cualquier tipo de dato (Ramírez, 2021). Las bases de datos clave-valor son adecuadas para aplicaciones que requieren un alto rendimiento para operaciones de lectura y escritura, como las aplicaciones de caché.

Almacenamiento documental

En este se almacena los datos como documentos. Los documentos pueden ser de cualquier formato, como JSON, XML o YAML. Las bases de datos documentales son adecuadas para aplicaciones que requieren un alto grado de flexibilidad, como las aplicaciones de gestión de contenido y las aplicaciones de análisis de datos (Ortega, 2022).

Almacenamiento columnar

Este tipo de almacenamiento organiza los datos por columnas, en lugar de por filas. Las bases de datos de familia de columnas son adecuadas para aplicaciones que requieren un alto rendimiento para operaciones de lectura y escritura de grandes cantidades de datos, como las aplicaciones de análisis de datos y las aplicaciones de big data (Sarasa, 2019).

Almacenamiento de grafos

Los registros se representan los datos como un grafo. Los nodos del grafo representan entidades y las aristas representan relaciones entre entidades (López, 2023). Las bases de datos de grafos son adecuadas para aplicaciones que requieren un alto rendimiento para operaciones de búsqueda y análisis de relaciones, como las aplicaciones de redes sociales y las aplicaciones de inteligencia artificial. Para el proyecto se utilizó este tipo de almacenamiento y la base de datos utilizadas fue Neo4j

Neo4j

Neo4j es una base de datos de grafos de código abierto implementada en Java. Es una base de datos totalmente transaccional y un motor Java persistente que almacena estructuras en forma de grafos en lugar de tablas (Scifo, 2020). En Neo4j, los datos se toman en forma de consulta y se escriben en el lenguaje Cypher. Es un lenguaje que está diseñado para admitir varias versiones de patrones en

los datos, lo que lo hace un lenguaje simple y lógico (Sholichah et al., 2020). Se eligió Neo4j, una base de datos de grafos de propiedades representativa, debido a su flexibilidad y eficiencia en la entrada y consulta de datos.

Lenguaje Cypher

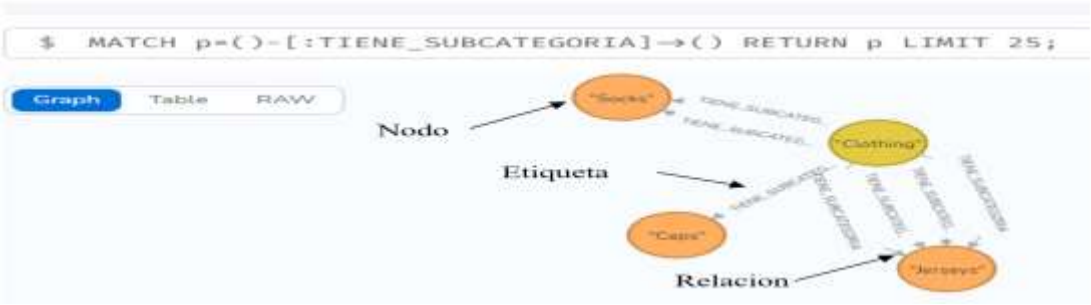
Cypher es el lenguaje de consulta gráfico y declarativo de Neo4j, de uso sencillo para desarrolladores que permite almacenar y recuperar datos de bases de datos gráficas, e incorpora las funciones de otros lenguajes estándar de acceso a datos. El lenguaje utiliza ASCII-Art para representar patrones gráficos visuales para encontrar o actualizar datos en Neo4j. Actualmente hay varias formas de ejecutar el lenguaje cypher para las consultas de Neo4j, por otra parte, Cypher no solo es la mejor forma de que los datos interactúen con

Neo4j, sino que también es de código abierto, lo que es mucho más sencillo que escribir consultas complejas de programas Java para acceder a la base de datos. Aunque se basa en la funcionalidad de SQL (Cao, 2023). Está optimizado específicamente para gráficos y tiene una sintaxis clara y concisa, lo que permite a los usuarios escribir fácilmente todas las operaciones CRUD normales de forma sencilla y mantenible. Además, proporciona consultas avanzadas, que incluyen agregación de datos (similar a SQL), consultas de funciones y consultas encadenadas.

Esquema Neo4j

El esquema que maneja Neo4j es mediante los nodos, relaciones y las restricciones, estas última según Webber y Bruggen (2020) son las propiedades de las etiquetas que identifican las relaciones entre nodos, los esquemas que se reflejan al realizar consultas son las siguientes:

Figura 1. *Esquemas en Neo4j*



Fuente: Elaboración propia

METODOLOGÍA

Para la investigación se realizó los siguientes pasos:

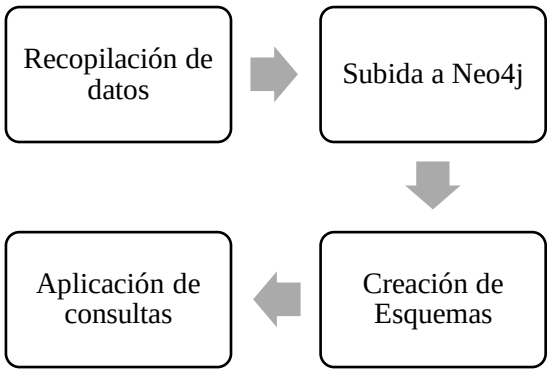


Figura *2.* Fases para el análisis de datos en grafos

Fuente: Elaboración propia

Para el desarrollo de la investigación se utilizó Neo4j y el el dataset que constaba de 30000 registros obtenidos de Kaggle perteneciente Flipkart(https://www.kaggle.com/datasets/aaditshukla/flipkart-fasion-products-dataset)

Recopilación de Datos

Como se mencionó para el desarrollo de la investigación se obtuvo el dataset de Kaggle el cual constaba de 30000 registros, y estaba compuesto de los siguientes campos:

Tabla 1.

Descripción de campos del dataset de ventas

Campo	Descripción	Tipo de dato
_id	Código del producto	String
actual_price	Precio actual	Decimal
average_rating	Calificación	String
Brand	Marca	String
Category	Categoría	String
crawled_at	Fecha de venta	String
description	Porcentaje de descuento	String

Discount	Descripción del descuento	String
Images	URL de las imágenes	String
out_of_stock	Fuera de stock	Booleano
Pid	Código del tracking	String
product_details	Detalles del producto	String
Seller	Vendedor	String
selling_price	Precio de Venta	Decimal
sub_category	Subcategoría	String
Title	Título asociado	String
url	URL del producto	String

Fuente: Elaboración propia

El dataset consta de 17 campos, 16 campos son de tipo String, dos de tipo Decimal y uno de tipo Booleano, los datos se encontraban en dos formatos .json y .xlsx, para este caso se utilizó el archivo de Excel y se procedió a la transformación a .csv para la manipulación y subida de los datos.

Subida a Neo4j

De todos los campos subidos se procedió a crear 8 nodos ya que eran los que eran los más relevantes para el estudio para crear el esquema siguiente:

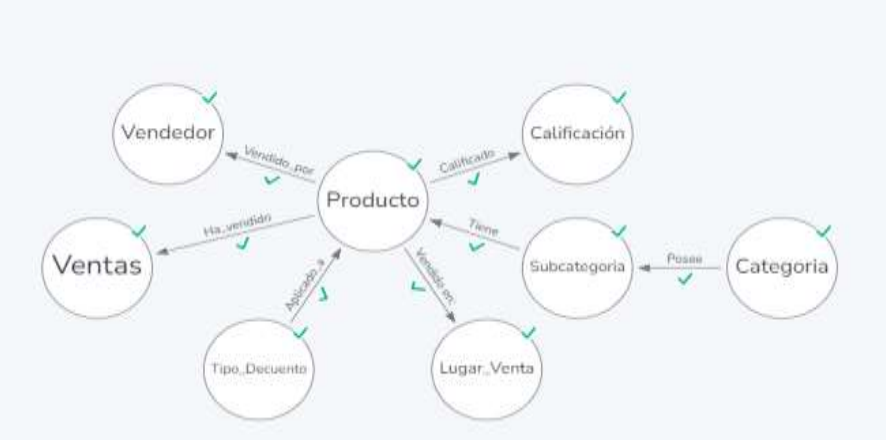


Figura *3.* Esquema de relaciones entre nodos

Fuente: Elaboración propia

Para la subida d ellos datos se aplicó las siguientes consultas en Cypher:

Tabla 2.

Descripción de campos del dataset de ventas

Nodo	Query generado
Producto	Key statement CREATE CONSTRAINT `imp_uniq_Producto__id` IF NOT EXISTS FOR (n: `Producto`) REQUIRE (n.`_id`) IS UNIQUE; Load statement UNWIND $nodeRecords AS nodeRecord WITH * WHERE NOT nodeRecord.`_id` IN $idsToSkip AND NOT nodeRecord.`_id` IS NULL MERGE (n: `Producto` { `_id`: nodeRecord.`_id` });
Vendedor	Key statement CREATE CONSTRAINT `imp_uniq_Vendedor_seller` IF NOT EXISTS FOR (n: `Vendedor`) REQUIRE (n.`seller`) IS UNIQUE; Load statement UNWIND $nodeRecords AS nodeRecord WITH * WHERE NOT nodeRecord.`seller` IN $idsToSkip AND NOT nodeRecord.`seller` IS NULL MERGE (n: `Vendedor` { `seller`: nodeRecord.`seller` });
Ventas	Key statement CREATE CONSTRAINT `imp_uniq_Ventas_actual_price` IF NOT EXISTS FOR (n: `Ventas`) REQUIRE (n.`actual_price`) IS UNIQUE; Load statement UNWIND $nodeRecords AS nodeRecord WITH * WHERE NOT nodeRecord.`actual_price` IN $idsToSkip AND NOT nodeRecord.`actual_price` IS NULL MERGE (n: `Ventas` { `actual_price`: nodeRecord.`actual_price` });
Tipo_Descuento	Key statement CREATE CONSTRAINT `imp_uniq_Tipo_Decuento_description` IF NOT EXISTS FOR (n: `Tipo_Decuento`) REQUIRE (n.`description`) IS UNIQUE; Load statement UNWIND $nodeRecords AS nodeRecord WITH *

	WHERE NOT nodeRecord.`description` IN $idsToSkip AND NOT nodeRecord.`description` IS NULL MERGE (n: `Tipo_Decuento` { `description`: nodeRecord.`description` });
Calificación	Key statement CREATE CONSTRAINT `imp_uniq_Calificación_average_rating` IF NOT EXISTS FOR (n: `Calificación`) REQUIRE (n.`average_rating`) IS UNIQUE; Load statement UNWIND $nodeRecords AS nodeRecord WITH * WHERE NOT nodeRecord.`average_rating` IN $idsToSkip AND NOT toFloat(trim(nodeRecord.`average_rating`)) IS NULL MERGE (n: `Calificación` { `average_rating`: toFloat(trim(nodeRecord.`average_rating`)) });
Subcategoría	Key statement CREATE CONSTRAINT `imp_uniq_Subcategoria_sub_category` IF NOT EXISTS FOR (n: `Subcategoria`) REQUIRE (n.`sub_category`) IS UNIQUE; Load statement UNWIND $nodeRecords AS nodeRecord WITH * WHERE NOT nodeRecord.`sub_category` IN $idsToSkip AND NOT nodeRecord.`sub_category` IS NULL MERGE (n: `Subcategoria` { `sub_category`: nodeRecord.`sub_category` });
Categoría	Key statement CREATE CONSTRAINT `imp_uniq_Categoria_category` IF NOT EXISTS FOR (n: `Categoria`) REQUIRE (n.`category`) IS UNIQUE; Load statement UNWIND $nodeRecords AS nodeRecord WITH * WHERE NOT nodeRecord.`category` IN $idsToSkip AND NOT nodeRecord.`category` IS NULL MERGE (n: `Categoria` { `category`: nodeRecord.`category` });
Lugar Venta	Key statement CREATE CONSTRAINT `imp_uniq_Lugar_Venta_brand` IF NOT EXISTS FOR (n: `Lugar_Venta`) REQUIRE (n.`brand`) IS UNIQUE; Load statement UNWIND $nodeRecords AS nodeRecord WITH *

	WHERE NOT nodeRecord.`brand` IN $idsToSkip AND NOT nodeRecord.`brand` IS NULL MERGE (n: `Lugar_Venta` { `brand`: nodeRecord.`brand` });

Fuente: Elaboración propia

Al subir los datos se creó todos los nodos visualizándose de la siguiente forma:

Figura *4.* Nodos creados

Fuente: Elaboración propia

Creación de Esquemas

Luego se procedió a la creación de relaciones estableciéndose las siguiente:

Tabla 3.

Creación de relaciones entre nodos mediante lenguaje Cypher

Nodos relacionados	**Nombre de la relación**	**Query Ejecutado**
Tipo_Descuento con Producto	Aplicado a:	UNWIND $relRecords AS relRecord MATCH (source: `Tipo_Decuento` { `description`: relRecord.`description` }) MATCH (target: `Producto` { `_id`: relRecord.`_id` }) MERGE (source)-[r: `Aplicado_a`]->(target);
Producto con Calificacion	Calificado	UNWIND $relRecords AS relRecord MATCH (source: `Producto` { `_id`: relRecord.`_id` }) MATCH (target: `Calificación` { `average_rating`: toFloat(trim(relRecord.`average_rating`)) })

		MERGE (source)-[r: `Calificado`]->(target);
Producto con Ventas	Ha_vendido	UNWIND $relRecords AS relRecord MATCH (source: `Producto` { `_id`: relRecord.`_id` }) MATCH (target: `Ventas` { `actual_price`: relRecord.`actual_price` }) MERGE (source)-[r: `Ha_vendido`]->(target);
Categoria con Subcategoria	Posee	UNWIND $relRecords AS relRecord MATCH (source: `Categoria` { `category`: relRecord.`category` }) MATCH (target: `Subcategoria` { `sub_category`: relRecord.`sub_category` }) MERGE (source)-[r: `Posee`]->(target);
Subcataegoría con Productos	Tiene	UNWIND $relRecords AS relRecord MATCH (source: `Subcategoria` { `sub_category`: relRecord.`sub_category` }) MATCH (target: `Producto` { `_id`: relRecord.`_id` }) MERGE (source)-[r: `Tiene`]->(target);
Lugar_Ventas con productos	Vendido en:	UNWIND $relRecords AS relRecord MATCH (source: `Producto` { `_id`: relRecord.`_id` }) MATCH (target: `Lugar_Venta` { `brand`: relRecord.`brand` }) MERGE (source)-[r: `Vendido en:`]->(target);
Vendedor con producto	Vendido por:	UNWIND $relRecords AS relRecord MATCH (source: `Producto` { `_id`: relRecord.`_id` }) MATCH (target: `Vendedor` { `seller`: relRecord.`seller` }) MERGE (source)-[r: `Vendido_por`]->(target);
Categoria_Ventas	Ha_vendido:segcategoria	UNWIND $relRecords AS relRecord MATCH (source: `Categoría` { `category`: relRecord.`category` }) MATCH (target: `Ventas` { `actual_price`: relRecord.`actual_price` }) MERGE (source)-[r: `ha_vendido_segcategoria`]->(target);
	Calificación categoría	UNWIND $relRecords AS relRecord MATCH (source: `Categoría` { `category`: relRecord.`category` })

Categoría y calificación		MATCH (target: `Calificación` { `average_rating`: toFloat(trim(relRecord.`average_rating`)) }) MERGE (source)-[r: `Calificación_categoría`]->(target);
Vendedor con calificación	obtenidocalificacionvendedor	UNWIND $relRecords AS relRecord MATCH (source: `Vendedor` { `seller`: relRecord.`seller` }) MATCH (target: `Calificación` { `average_rating`: toFloat(trim(relRecord.`average_rating`)) }) MERGE (source)-[r: `obtenidocalificacionvendedor`]->(target);

Fuente: Elaboración propia

Como resultado se obtuvo las siguientes relaciones creadas:

Figura 5. Visualización de nodos y relaciones

Fuente: Elaboración propia

Aplicación de Consultas

Una vez que se obtuvo el esquema y las relaciones se procedió a la generación de las siguientes consultas:

Tabla 4.

Creación de consultas para la obtención de datos relevantes

Nodo	Query generado
Descuento aplicado	MATCH (d:Tipo_Decuento)-[:Aplicado_a]->(p:Producto) WITH d.description AS Descuento, COUNT(p._id) AS Cantidad ORDER BY Cantidad DESC LIMIT 10 RETURN Descuento, Cantidad
Producto más vendido	MATCH (p:Producto)-[:Ha_vendido]->(v: Ventas) WITH p._id AS Producto, SUM(tofloat(v.actual_price)) AS Total ORDER BY Total DESC LIMIT 10 RETURN Producto, Total
Lugar donde se ha vendido más productos	MATCH p=()-[:`Vendido en:`]->() RETURN p LIMIT 1000;
Promedio de calificaciones	MATCH (p:Producto)-[:Calificado]->(c:Calificación) WITH p._id AS Producto, AVG(c.average_rating) AS Promedio ORDER BY Promedio DESC LIMIT 10 RETURN Producto, Promedio
Categoría más vendida	MATCH p=()-[:ha_vendido_segcategoria]->() RETURN p LIMIT 200;
Vendedor con más productos vendidos	MATCH p=()-[:Vendido_por]->() RETURN p LIMIT 30000;
Calificación por vendedor	MATCH (p:Vendedor)-[:obtenidocalificacionvendedor]->(c:Calificación) WITH p.seller AS Vendedor, AVG(c.average_rating) AS Promedio ORDER BY Promedio DESC LIMIT 10 RETURN Vendedor, Promedio
Calificación por categoría	MATCH (p:Categoría)-[:Calificación_categoría]->(c:Calificación) WITH p.category AS Categoría, AVG(c.average_rating) AS Promedio ORDER BY Promedio DESC LIMIT 10 RETURN Categoría, Promedio

Fuente: Elaboración propia

RESULTADOS

Al realizar la consulta de los descuentos aplicados sobre los productos se pudo evidenciar que el comúnmente utilizado fue el 60%, seguido del 50% a pesar que el 66% era el mayo este no fue muy aplicado en las compras.

Figura *6.* Resultados de los descuentos aplicados

Fuente: Elaboración propia

En cuanto s los datos relacionados al top 10 de productos no se tuvo datos de relevancia ya que el producto estaba identificado por su código y apareció los de las ventas igual a 999.00 rupias ya que esta es la moneda en India. Al consultar el lugar donde se vendió más productos se obtuvo el grafo, concentrado en el primer círculo marcado en el gráfico el cual pertenecía Fairdea.

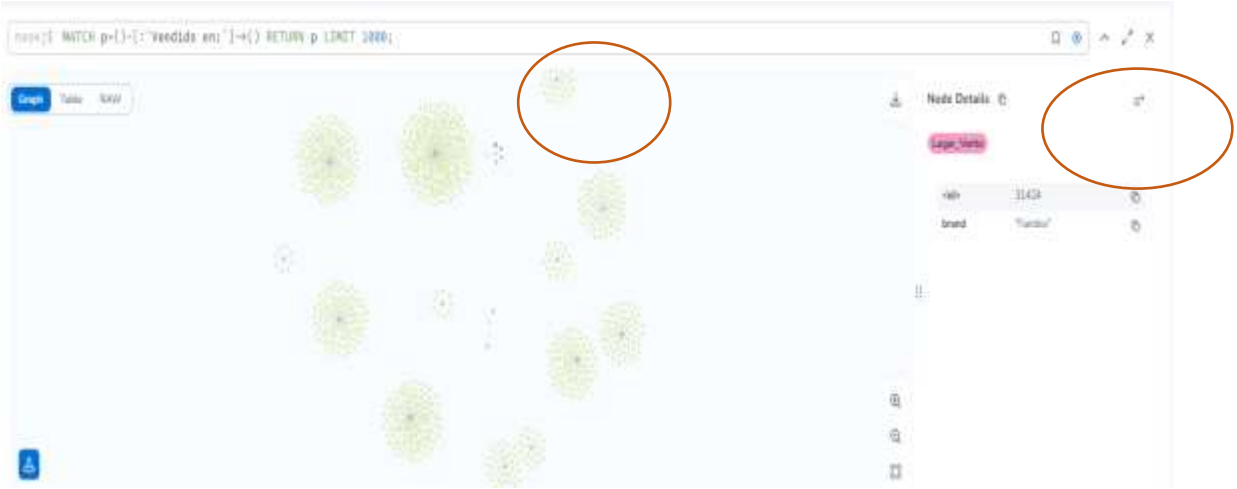


Figura *7.* Resultados de los promedios de productos

Fuente: Elaboración propia

Por otra parte, también se realizó la consulta sobre el promedio de calificaciones otorgados a un producto, teniendo lo siguiente:

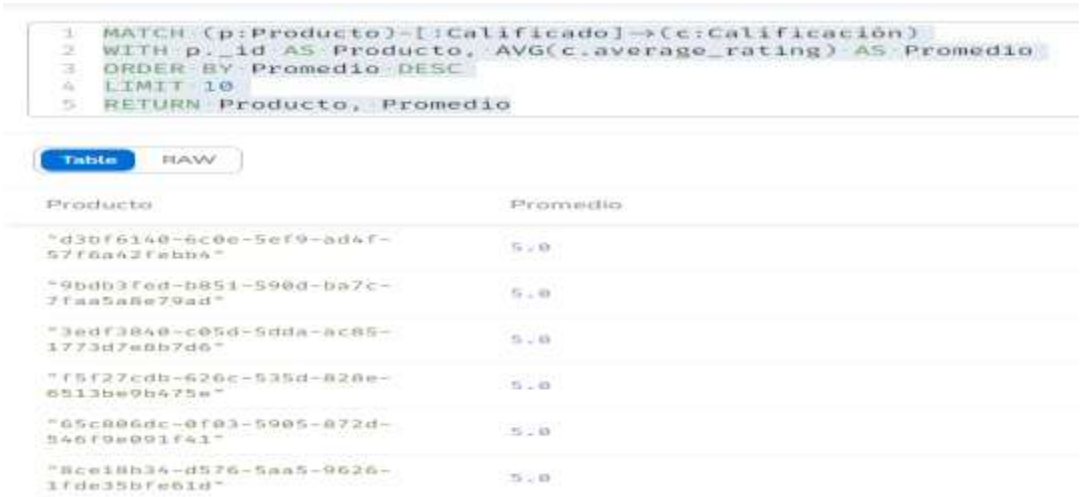


Figura *8.* Resultados de los promedios de productos

Fuente: Elaboración propia

También se hizo la consulta de la categoría más vendida el en la gráfica se pudo evidencia que fue Clothing and Accesories, seguido de "Footwear","Bags, Wallets & Belts" y "Toys"

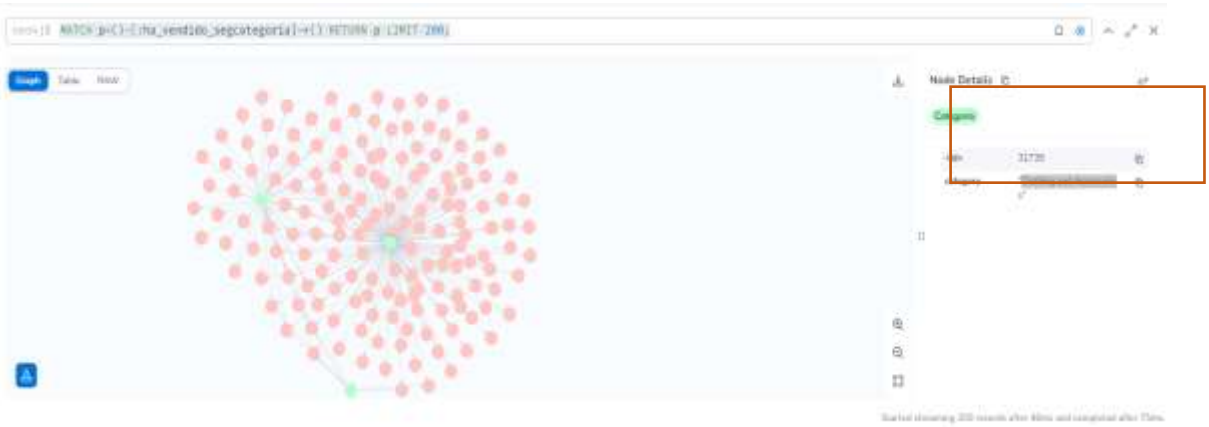


Figura *9.* Resultados de los promedios de productos

Fuente: Elaboración propia

A pesar que la gráfica no se puede visualizar de forma específica en la parte izquierda se visualiza este detalle.

Además, el vendedor al cual le compararon más productos según la consulta realizada fue "InstaHeaven".

4

Figura *10.* Resultados de los promedios de productos

Fuente: Elaboración propia

Y el vendedor con calificación más alta fue "VARTees(Not Enough Ratin"

```
MATCH (p:Vendedor)-[:obtenidocalificacionvendedor]->(c:Calificacion)
WITH p.seller AS Vendedor, AVG(c.average_rating) AS Promedio
ORDER BY Promedio DESC
LIMIT 10
RETURN Vendedor, Promedio
```

Table | RAW

Vendedor	Promedio
"VARTees(Not Enough Ratin"	5.0
"MILDINC(New Sell"	5.0
"RungExclusive(Enough Ratin"	5.0
"Ruffike"	4.7
"MOM AND SON"	4.7
"Thebesta"	4.6

Figura *11.* Resultados de los promedios de calificación según vendedores

Fuente: Elaboración propia

También se verificó a la calificación por categoría obteniendo los siguientes resultados:

```
MATCH (p:Categoria)-[:Calificación_categoria]->(c:Calificación)
WITH p.category AS Categoria, AVG(c.average_rating) AS Promedio
ORDER BY Promedio DESC
LIMIT 10
RETURN Categoria, Promedio
```

Table | RAW

Categoría	Promedio
"Bags, Wallets & Belts"	3.9166666666666667
"Toys"	3.6
"Footwear"	3.437931034482759
"Clothing and Accessories"	3.1394736842105266

Figura *12.* Resultados de los promedios de calificación según categoría

Fuente: Elaboración propia

Concluyendo que a pesar de que Clothing and accesories vende más, el mayor evaluado es Bags, Wallets&Belts.

DISCUSIÓN

El lenguaje Cypher es mucho más sencillo que el lenguaje Sql ya que permite que las consultas sean visualizadas de forma gráfica, por lo cual en el estudio desarrollado por Carrillo y Galpin (2021) analizaron datos pertenecientes a Mercado Libre identificando las categorías más importantes y los usuarios más influyentes. De forma similar en esta investigación se pudo identificar las categorías de productos más vendidas y las ofertas más utilizadas por parte de los usuarios.

A pesar que las bases de datos relacionales, el análisis y su diseño requiere mucho tiempo y es costoso. La representación de datos en grafos gráficos proporciona un enfoque más flexible y tiene como objetivo encontrar representaciones eficientes, lo cual fue destacado por los autores Walke et al (2023) quienes aplicaron bases de datos basados en grafos para el análisis de datos clínicos verificando síntomas comunes entre pacientes, verificando tal como el presente estudio que estas permiten un análisis profundo siempre y cuando los datos se encuentren relacionados entre sí .

CONCLUSIONES

A diferencia de las bases de datos tradicionales las orientadas a grafos permiten verificar la información de forma gráfica lo que permite a cualquier usuario realizar un análisis rápido de los datos encontrando tendencias. El análisis de la base de datos orientada a grados permitió obtener información valiosa sobre las ventas, los vendedores y los lugares de venta en la plataforma de e-commerce.

Esta información puede ser utilizada por la empresa para mejorar sus estrategias de marketing y optimizar el rendimiento de la plataforma. Para realizar este tipo de análisis en cuanto a productos es necesario conocer el nombre de estos, sin embargo, en el dataset sólo mostraba el código perteneciente a este lo cual dificultó un poco conocer a profundidad las características a menos que se consultara todos los detalles en el archivo de origen.

Además, es recomendable que para realizar las consultas se evalúe el esquema y relaciones posibles entre los nodos para que se obtenga los datos de forma confiable.

REFERENCIAS BIBLIOGRÁFICAS

- Arenado, S. (2023). *Analítica de redes y grafos.* Trabajo Fin de Grado, Universidad de Sevilla,Sevilla.https://idus.us.es/bitstream/handle/11441/148026/TFG4529_Arenado%20Serrano.pdf?sequence=1&isAllowed=y

- Cao, J. (2023). *E-Commerce Big Data Mining and Analytics.* Hefei, Springer. https://doi.org/ISBN:9789819935888, 9819935881

- Carrillo, G., & Galpin, I. (2021). Analítica de grafos para identificar entidades relevantes y comunidades en Mercado Libre: un estudio de caso. *MUTIS*, 77-95. https://doi.org/https://doi.org/10.21789/22561498.1740

- López, V. (2023). *Sistemas de Big Data.* Ra-Ma S.A. Editorial y Publicaciones. https://doi.org/ISBN:9788419857446, 8419857440

- Marrero, L., Verena, O., Tesone, F., Thomas, P., Corbalán, L., Fernández, J., & Pesado, P. (2023). Un Estudio de Procesos de Diseño de Bases de Datos NoSQL. (U. N. Plata, Ed.) *Sedici*,404-414. http://sedici.unlp.edu.ar/bitstream/handle/10915/149452/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y

- Migani, S., Vera, C., & Lund, M. J. (26 de Abril de 2018). NoSQL_ Modelo de datos y sistemas de gestión de base de datos. *RedUNCI - UNNE*, 225-228. https://doi.org/ISBN 978-987-3619-27-4

- Ortega, J. (2022). *Big data, machine learning y data science en python.* RA-MA S. A. EditorialyPublicaciones.https://www.google.com.ec/books/edition/Big_data_machine_learning_y_data_science/81W6EAAAQBAJ?hl=es419&gbpv=1&dq=tipos+de+base+de+datos+NOSQL&pg=PT59&printsec=frontcover

- Peirano, J., Rojas, J., & Romero, A. (2021). *Uso de Neo4J como base de datos orientada a grafos para la gestión de información de paradas, recorridos y viajes del Sistema de Transporte Metropolitano (STM) de la Ciudad de Montevideo, Uruguay.* Trabajo de Titulación, Universidad ORT Uruguay, Facultad de Ingeniería, Montevideo.

https://dspace.ort.edu.uy/bitstream/handle/20.500.11968/6393/Material%20completo.pdf?sequence=1&isAllowed=y

- Ramírez, Ó. (2021). *Python a fondo.* Marombo. https://doi.org/ISBN:9788426732873, 8426732879
- Saorín, T. (01 de Diciembre de 2019). Grafos de conocimiento y bases de datos en grafo: conceptos fundamen-tales a partir de una 'obra maestra' del Museo del Prado. *Anuario ThinkEPI, 13*. https://doi.org/https://doi.org/10.3145/thinkepi.2019.e13f05
- Sarasa, A. (2019). *Introducción a las bases de datos NSQL clave-valor usando Redis.* EditorialUOC,S.L.https://www.google.com.ec/books/edition/Introducci%C3%B3n_a_las_bases_de_datos_NSQL/Q6i8DwAAQBAJ?hl=es419&gbpv=1&dq=tipos+de+base+de+datos+NOSQL+columnas&pg=PT19&printsec=frontcover
- Scifo, E. (2020). *Hands-On Graph Analytics with Neo4j*. Birmingham: Packt Publishing. https://www.google.com.ec/books/edition/Hands_On_Graph_Analytics_with_Neo4j/Dyj5DwAAQBAJ?hl=es-419&gbpv=1
- Sholichah, R. J., Imrona, M., & Alamsyah, A. (24 de Septiembre de 2020). Performance Analysis of Neo4j and MySQL Databases using Public Policies Decision Making Data. *IEEE*, 152-157. https://doi.org/10.1109/ICITACEE50144.2020.9239206
- Walke, D., Miguel, D., Schallert, K., Muth, T., Broneske, D., Saake, G., & Heyer, R. (2023). The importance of graph databases and graph learning for clinical applications. *PubMed*. https://doi.org/10.1093/database/baad045
- Webber, J., & Bruggen, R. V. (2020). *Graph databases for dummies.* New Jersey, Estados Unidos: John Wiley & Sons. https://doi.org/ISBN: 978-1-119-74602-7 (pbk); ISBN: 978-1-119-74579-2 (ebk)
- Wu, W. (13 de Marzo de 2021). https://dl.acm.org/doi/pdf/10.1145/3408877.3432541. *System A*, 590-596. https://dl.acm.org/doi/pdf/10.1145/3408877.3432541
- Yan, B., Yang, C., Shi, C., Fang, Y., Li, Q., Ye, Y., & Du, J. (2023). Graph Mining for Cybersecurity: A Survey. *ACM*. https://doi.org/https://dl.acm.org/doi/pdf/10.1145/3610228

- Zaki, M., & Meira, W. (2014). En M. Zaki, & W. Meira, *Data Mining and Analysis Fundamental Concepts and Algorithms* (pág. 93). Cambridge University Press. https://www.webpages.uidaho.edu/~stevel/517/Data%20Mining%20and%20Analysis%20by%20Zaki.pdf