Mejora en la Gestión de Aeropuertos en América mediante Base de Datos NoSQL

Manrique Camacho P.[1], Pamela Argueta D.[2], Josué Flores J.[3], Ashly Valerio S[4] y Johnson Montero S.[5]

[1]Escuela de Estadística, manrique.camacho@ucr.ac.cr
[2]Escuela de Estadística, pamela.argueta@ucr.ac.cr
[3]Escuela de Estadística, josue.flores@ucr.ac.cr
[4]Escuela de Estadística, ashly.valerio@ucr.ac.cr
[5]Escuela de Estadística, johnson.montero@ucr.ac.cr

***Resumen*—Este proyecto se enfoca en la gestión y análisis de datos relacionados con la industria de aeropuerto. Se utilizan dos sistemas de gestión de bases de datos, ArangoDB y OrientDB, para comparar su desempeño y facilidad de uso en la inserción, consulta y análisis de datos. El proyecto destaca las ventajas y problemáticas de cada sistema, así como las soluciones implementadas para garantizar la eficiencia en la manipulación de datos. Además, se evalúa la consistencia de los datos entre ambos sistemas y se resalta la importancia de la herramienta Python en la inserción de datos. En general, el proyecto busca mejorar la comprensión de la gestión de datos NoSQL. Se encontró que ArangoDB tiene una leve ventaja sobre OrientDB, sin embargo estas conclusiones recaen en el sistema utilizado de Python.**

***Palabras clave*—ArangoDB, OrientDB, Python, Web-Scrapping, NoSQL, Aeropuertos.**

I. Introducción

Los aeropuertos son una parte esencial de la infraestructura de transporte de América. Son los puntos de entrada y salida para millones de pasajeros y carga cada año. La gestión eficiente de los aeropuertos es fundamental para garantizar la seguridad, la eficiencia y la comodidad de los viajeros.Las bases de datos NoSQL son un tipo de base de datos que está diseñado para almacenar datos de forma flexible y escalable. Son una alternativa a las bases de datos relacionales tradicionales, que son más adecuadas para almacenar datos estructurados y es por ello que se utiliza para este análisis.

Esta mejora en la gestión tiene el potencial de impactar positivamente en varios aspectos clave, como la coordinación de vuelos, la seguridad, la experiencia del pasajero y la eficiencia operativa. Al adoptar tecnologías avanzadas para gestionar grandes volúmenes de datos y relaciones interconectadas, se busca proporcionar a los aeropuertos de América una herramienta robusta que facilite la toma de decisiones informada y promueva la eficacia en la operación diaria.

Para analizar la eficiencia de las bases NoSQL se exploran ArangoDB y OrientDB, ya que pueden contribuir a la mejora en la gestión de aeropuertos en América se presenta como una iniciativa de gran relevancia y actualidad. Esta elección tecnológica se alinea con los avances en bases de datos y la necesidad de optimizar la gestión en una industria crítica y en constante evolución. La gestión aeroportuaria se beneficiaría significativamente de la aplicación de estas tecnologías en busca de una operación más eficiente, segura y orientada al futuro en América.

II. Descripción del problema

La gestión eficiente de relaciones y conexiones en la industria de aeropuertos es importante para garantizar las mejores operaciones puesto que los aeropuertos cuentan con mucho flujo tanto de personas, personal como tránsito de aviones. Los aeropuertos tienen relaciones complejas con aerolíneas, proveedores, reguladores y otras entidades. La gestión de estas relaciones y datos es importante para la coordinación de vuelos y servicios. Este trabajo se centrará en la región del continente de América, sin embargo, en el futuro se podrá implementar a los demás continentes.

III. Objetivo del proyecto

El objetivo de este proyecto es desarrollar una solución que permita la gestión eficiente de relaciones en los aeropuertos de América utilizando una base de datos orientada a grafos. Se buscará mejorar la representación de relaciones y facilitar la consulta y navegación de información para la operación de

aeropuertos en el continente, para que de esta manera haya un mejor control y facilidad para los trabajadores del aeropuerto, como el control de tráfico aéreo, usuarios/pasajeros, aerolíneas, proveedores e incluso reguladores de aviación.

IV. Metodología

IV-A. Tipos de bases de datos

Para el trabajo, tal como lo menciona el título, se utilizarán bases de datos orientadas a grafos. Dichas bases son una plataforma especializada y de un solo propósito para crear y manipular datos de manera de grafos. Los grafos contienen nodos, bordes y propiedades que se utilizan para representar y almacenar datos de una forma que no permiten las bases de datos relacionales. En el análisis de grafos, los algoritmos se centran en explorar relaciones y distancias. Por ejemplo, para determinar la importancia, los algoritmos suelen fijarse en los bordes de entrada, la importancia de los vértices circundantes [1] Para encontrar una mejor eficiente de la gestión de aeropuertos se recurrio a utilizar web-scrapping, en el lenguaje de programación de Python, con la librería de Selenium y Bs4, entre otras... Dentro se consiguieron un total de 3 bases de datos, que consistían en una de vuelos, información del aeropuerto y aerolíneas. Aproximadamente se tienen un total de 16.000 filas, es decir que la base de datos tuvo un total de 16.000 nodos; además cada una de las bases contiene diferentes tamaños de columnas, estas serán las características que tendrá cada nodo. Ya teniendo una introducción de qué es el modelo de bases de datos orientados a grafos y una explicación de la recolección de la base de datos, se determinan los nodos correspondientes al trabajo de investigación. La primera relación sería entre aeropuertos y rutas, en donde los aeropuertos serían los nodos en el grafo y las rutas serían las conexiones, esta conexión llevaría información como el vuelo, el número de vuelo y la aerolínea. Seguidamente, es la relación entre aerolíneas y rutas, en donde las aerolíneas serían los nodos y las rutas operadas por cada aerolínea serían las conexiones, lo cual permitiría identificar qué aerolínea opera que rutas. La siguiente figura muestra esta conexión que se espera hacer.

Este estudio de las bases NoSQL en el ámbito de aviación no ha sido casi abordado, sin embargo, hay un análisis en el cual utilizan Neo4j para un análisis de rutas. En donde, el proyecto consistía en crear todas las rutas de conexión en los aeropuertos. Esta investigación fue realizada por Muchow, Florian & Conzelmann (2017) [2]. Por lo cual esta investigación ayudará como referencia a desarrollar el proyecto.

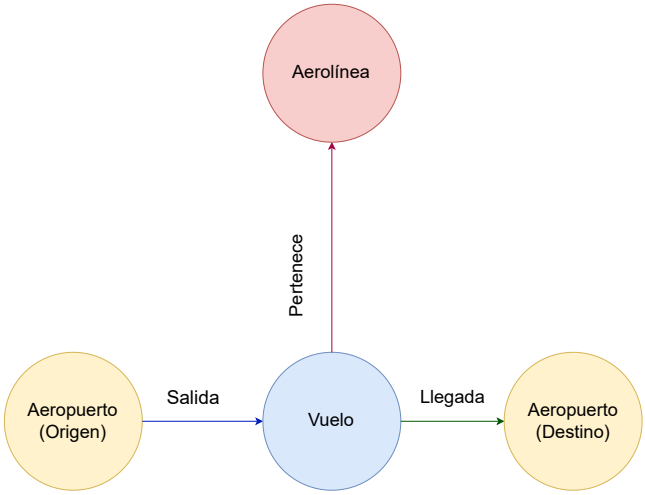

Figura 1. Estructura de Base.

IV-B. Software

La creación de las bases de datos orientadas a grafos serán implementadas en los siguientes softwares:OrientDB y ArangoDB.

IV-B1. OrientDB: Es una base de datos NoSQL de código abierto, la cual fue desarrollada en JAVA. Esta base de datos acepta varios modelos de datos, entre ellos el modelo orientado a grafos. El software, OrientDB, tiene varias ventajas con respecto a otros softwares que manejan bases de datos orientadas a grafos. La primera es la velocidad, esta base de datos puede almacenar hasta 120,000 registros por segundo y procesar transacciones a una velocidad 10 veces superior a la de sus competidores. Además, ofrece una solución de base de datos para usar en cualquier plataforma y utiliza SQL como lenguaje de consulta, lo que permite obtener resultados de manera rápida. Por último, OrientDB ahorra en costos al eliminar la necesidad de adquirir múltiples licencias y al brindar acceso a su documentación, es decir ser open source [3]. Por estas razones se ha decidido utilizar OrientDB como

software para la investigación.

Para comenzar a trabajar con OrientDB, se debe crear una base de datos. Esto se puede realizar mediante la CLI o programáticamente a través de las APIs proporcionadas. La CLI permite definir las características de la base de datos, como el tipo de almacenamiento (documento, gráfico, entre otros).

Desde la interfaz de inicio permite crear una base de datos en este caso es aeropuertos. En la cual se crearon los nodos y aristas que permitieron relacionar los datos en forma de grafo. La inserción de los datos se realizó mediante lenguaje SQL al igual que la creación de los nodos y aristas.

Para poder visualizar la base de datos de grafos se debe hacer una consulta en el apartado de GRAPH, dicha consulta se va realizar en forma de gráfico como se muestra en lo siguiente

IV-B2. ArangoDB: ArangoDB fue fundada en 2015 en Colonia, Alemania, que desarrolla la plataforma ArangoGraph Insights. ArangoDB se utiliza como base escalable para el análisis de datos en grafos y arquitecturas de datos complejas en miles de grandes empresas (Fortune 500) y nuevas empresas innovadoras de diversas industrias, como servicios financieros, atención médica y telecomunicaciones [4]. ArangoDB al igual que OrientDB tiene una gran cantidad de ventajas entre ellas resaltan: la consolidación del software, escalabilidad de rendimiento simplificada, reducción de la complejidad operativa, entre otros... Para trabajar con ArangoDB, se puede optar por dos enfoques: desde la línea de comandos la interfaz de usuario web o con Python, con la libreria de python-arango. En esta investigación, exploramos el uso de la interfaz web para la creación de una base de datos llamada AeropuertosDB junto con sus respectivas colecciones o nodos. Además se utilizo python-arango para comparar con la librería de pyOrient, en donde se evaluaron la inserción de datos como la interfaz y facilidad de los comandos de cada libreria.

Dentro de esta base de datos, se establecieron tres nodos, uno que contiene información básica sobre los aeropuertos, otro para las aerolíneas y un tercero para registrar los vuelos. Además, se establecen dos relaciones, denominadas Salen, Llegan y Pertenece, tal como se muestra en la figura 1. A diferencia de OrientDB para consultar los datos almacenados, se utiliza el lenguaje de consulta AQL. Este es un lenguaje declarativo que facilita la realización de consultas complejas en la base de datos [5].

Para gestionar la conexión entre ArangoDB y el servidor, se ha empleado Python como una herramienta eficaz. Para visualizar la base de datos de manera gráfica, se utiliza el apartado GRAPH.

V. Solución implementada con cada producto

V-A. Solución implementada con OrientDB

El uso de OrientDB, ha sido el mismo que se utilizó para ArangoDB. En el cuál pusimos a prueba tanto con una insercion de datos muy grandes como pequeña al programa. Las pruebas de inserción de datos grandes se intentaron hacer con la libreria de PyOrient, sin embargo esta libreria al estar tan desactualizada, presentaba una gran cantidad de errores. De esta forma se utilizó un bot, con ayuda de la libreria de Selenium, en el cuál insertaba los datos. Por otro lado para la inserción manual se probo con una cantidad de datos pero no tan grandes.

V-B. Solución implementada con ArangoDB

La implementación de ArangoDB siguió un enfoque similar al utilizado para OrientDB en el proyecto. Se realizaron pruebas exhaustivas, tanto con conjuntos de datos masivos como más pequeños, para evaluar su rendimiento.

Para la inserción de datos a gran escala, se optó por la librería PyArango, utilizada en OrientDB, presentaba problemas significativos. En lugar de PyOrient, se implementó un bot que utilizaba Selenium para automatizar la inserción de datos en ArangoDB. Esta estrategia resultó eficiente y confiable.

En las pruebas manuales de inserción de datos más pequeños en ArangoDB, no se detectaron problemas significativos. Se destacó que la librería Python para ArangoDB está actualizada y bien mantenida, lo que facilitó una experiencia de desarrollo sin complicaciones y una gestión eficaz de datos en ambos escenarios.

VI. Características de los Productos

VI-A. Consistencia

En términos de consistencia, ambos sistemas se destacan, ya que admiten transacciones ACID, lo que garantiza la integridad de los datos en situaciones críticas, como la coordinación de vuelos y servicios en aeropuertos.

Ambos sistemas demostraron mantener una consistencia alta. Durante las pruebas y comparaciones, no se identificaron problemas en la consistencia de los datos almacenados en ambas bases de datos. Incluso en situaciones de inserción y recuperación de datos concurrentes, los sistemas aseguraron que los datos reflejaran un estado coherente y correcto en todo

momento. Nunca se presentaron fallas ni en las funciones de los gráficos ni consultas

Es importante destacar que, a pesar de la alta consistencia general, se identificaron algunas fallas en las operaciones de inserción de datos en OrientDB cuando se utilizó la biblioteca de Python PyOrient, que no ha sido actualizada durante los últimos seis años. Esto se describen con más detalle en la sección de ”Desempeño en inserción de datos”. Sin embargo, es fundamental mencionar que estas deficiencias no afectaron la consistencia de los datos ya insertados, pero indican limitaciones específicas de la biblioteca PyOrient.

Sin embargo, en términos de consistencia tanto ArangoDB como OrientDB son opciones muy buenas, y no se tiene una ventaja por ninguno de los dos

VI-B. Desempeño en inserción de datos

El desempeño de la inserción de datos en el proyecto fue medido con distintas pruebas, las cuales se llevaron a cabo de la siguiente manera:

1. **Inserción de Datos de Grandes Cantidades**: Para simular un entorno de producción real, se realizó la inserción masiva de datos en ambas bases de datos. ArangoDB mostró un desempeño excepcional en este aspecto. La librería Python para ArangoDB facilitó la inserción eficiente de grandes conjuntos de datos. En contraste, OrientDB presentó problemas al utilizar PyOrient para esta tarea y requirió una solución alternativa basada en Selenium.
2. **Inserción Manual de Datos**: Se realizaron pruebas de inserción manual con conjuntos de datos más pequeños para evaluar la facilidad de uso y el rendimiento en condiciones menos demandantes. Ambos sistemas demostraron eficacia en esta categoría.
3. **Comparación de Rendimiento**: Se comparó el rendimiento de ambas bases de datos en cuanto a tiempos de inserción, estabilidad y escalabilidad. ArangoDB destacó por su capacidad para manejar grandes cantidades de datos sin sacrificar la velocidad, mientras que OrientDB mostró dificultades con PyOrient en situaciones de alta demanda.

En resumen, ArangoDB demostró ser la mejor elección para la inserción de datos en este proyecto debido a su desempeño, la disponibilidad de una librería Python bien actualizada y la capacidad de manejar grandes conjuntos de datos sin complicaciones. La implementación en ArangoDB resalta la importancia de seleccionar la tecnología adecuada según las necesidades específicas del proyecto y su escalabilidad.

VI-C. Desempeño en recuperación de datos

En general, tanto ArangoDB como OrientDB presentan un buen desempeño en la recuperación de datos. Ambos sistemas son altamente eficientes al ejecutar consultas y recuperar información de las bases de datos. Sin embargo, es importante destacar que se detectaron algunas diferencias notables cuando se utilizó Python para interactuar con los sistemas.

Mientras que ArangoDB mantuvo su eficiencia, OrientDB experimentó ciertas limitaciones debido a la necesidad de utilizar Selenium en conjunto con Python para algunas tareas, lo que lo limitó más. En resumen, la recuperación de datos en ambos sistemas fue buena, pero se observaron diferencias notables cuando se empleó Python para interactuar con OrientDB.

VI-D. Facilidad de uso

En términos de facilidad de uso, ArangoDB tiene una ventaja destacada. Ofrece una curva de aprendizaje más suave para los usuarios gracias a su modelo de datos multimodelo, lo que permite trabajar con documentos, gráficos y colecciones de clave-valor en un solo sistema. Además, su lenguaje de consulta, AQL, es intuitivo y potente. Por otro lado, OrientDB, aunque versátil, puede resultar más complejo debido a su orientación a gráficos y su consulta SQL, lo que puede requerir un tiempo adicional para familiarizarse. En este aspecto, ArangoDB proporciona una experiencia más amigable y accesible para los usuarios.

VI-E. Herramientas de gestión

ArangoDB y OrientDB, ofrecen herramientas de gestión para facilitar la administración de sus bases de datos.

OrientDB y ArangoDB son capaces de gestionar datos en un modelo de gráficos, lo que lo hace especialmente adecuado para aplicaciones que requieren un alto grado de relaciones y conexiones entre datos. Sin embargo, es importante mencionar que OrientDB presenta ciertas limitaciones al interactuar con Python, ya que se requiere la utilización de Selenium para algunas tareas, lo que podría complicar la gestión y administración de datos en comparación con ArangoDB. Observar la figura 2, la cual muestra el gráfico del aeropuerto de Chicago como nodo principal. Por otro lado, si se busca ver el gráfico realizado en OrientDB, ver figura 3

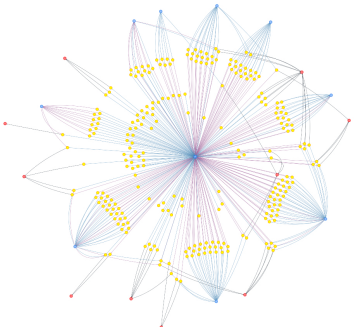

Figura 2. Gráfico de grafos en ArangoDB.

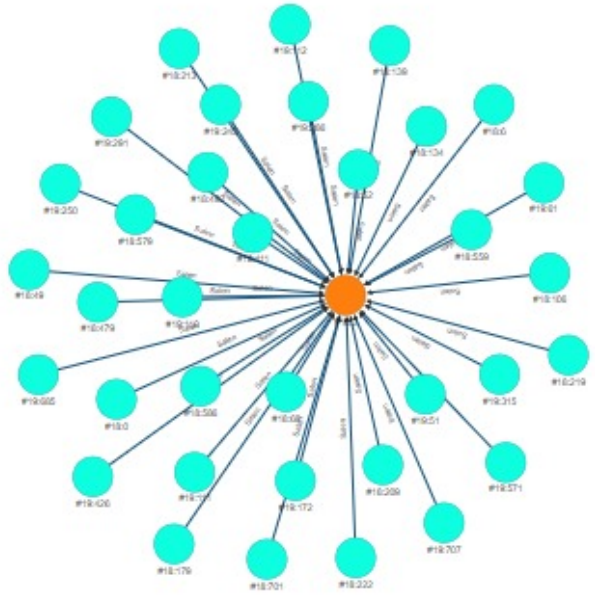

Figura 3. Gráfico de grafos en OrientDB.

En el contexto del proyecto, se emplearon estas herramientas para la administración de bases de datos, junto con Python como una solución para la inserción de datos, lo que permitió aprovechar las ventajas de ambos sistemas en términos de modelado y gestión de datos.

VI-F. Método utilizado para hacer consultas

Durante la fase de consulta de datos, se compararon las capacidades de ArangoDB y OrientDB. ArangoDB utiliza AQL (ArangoDB Query Language), un lenguaje flexible que

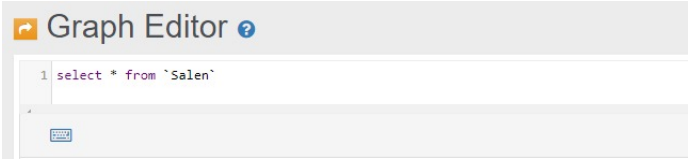

Figura 4. Consulta en Orient SQL.

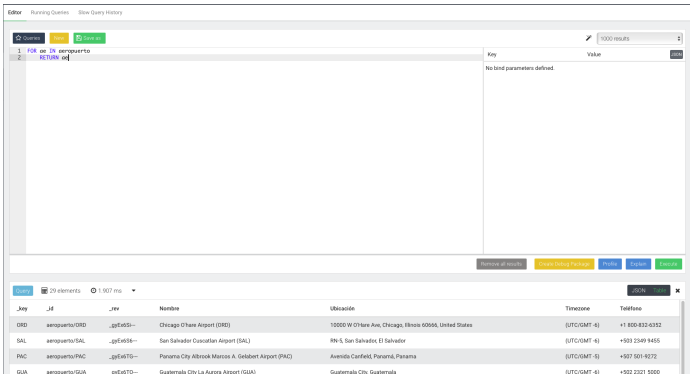

Figura 5. Consulta en AQL.

se adapta bien a datos en formato JSON y permite consultas avanzadas. Por otro lado, OrientDB emplea Orient SQL, optimizado para su modelo de datos orientado a grafos. Ambos sistemas son eficientes en sus respectivas áreas, con ArangoDB destacando en consultas sobre datos JSON y OrientDB en estructuras de grafo interconectadas. Ver figura 4 y 5

VII. Conclusiones

En conclusión, al comparar OrientDB y ArangoDB se ha demostrado que cada una tiene sus ventajas y sus desventajas, no obstante desde el analisis realizado en esta investigación, se demostró que ligeramente ArangoDB tiene una ventaja sobre OrientDB. Primeramente, por la facilidad de insercion de datos y que no presentaba ningún problema de este. Sin embargo aparte de ese fallo que OrientDB sostiene, los dos productos son muy similares, en terminos de desempeño general. Esta ventaja adjudicada a ArangoDB recae en que el equipo utiliza Python para la insercion de datos, pero si esta fuera manualmente, se diria que ambos son iguales y intercambiales entre ellos. Cabe resaltar la diferencia de OrientDB que puede traer mas usuarios, el cual es el uso de SQL como lenguaje de consulta y no el de ArangoDB el cual es AQL, muy utilizado por diferentes softwares de bases de datos de grafos.

VIII. Repositorio de Github

https://github.com/ManriCamachoP/AeropuertoDB

Referencias

[1] O. México, "¿qué es una base de datos orientada a grafos?" s.f. [Online]. Available: https://www.oracle.com/mx/autonomous-database/what-is-graph-database/

[2] F. Muchow, M. Conzelmann, and Y. Sagr, "Nosql use case: Airline reservation system," 2017.

[3] OrientDB, "Overview of orientdb," s.f. [Online]. Available: http://orientdb.org/docs/3.0.x/misc/Overview.html

[4] ArangoDB, “About us,” 2023. [Online]. Available: https://www.arangodb.com/about-arangodb-2/

[5] J. S. M. Support, “Using assets query language (aql) syntax,” s.f. [Online]. Available: https://support.atlassian.com/jira-service-management-cloud/docs/use-assets-query-language-aql/