CSP 554 — Big Data Technologies

Project Draft - Group - 4

Project topic #1 - Explore use of cloud NoSQL databases in depth

DocumentDB vs. MongoDB: Comparing NoSQL Databases

Team Members:
Sailavanya Narthu, A20516764
Arundhathi Nalla , A20549875
Sai Sandeep Neerukonda, A20550565

Introduction

In the domain of modern data management, our project delves into the exploration of Cloud NoSQL Databases, gaining inspiration from the ever-evolving environment of Big Data Technologies. NoSQL databases, which depart from standard tabular architectures, introduce an approach in which databases are not limited to tables and SQL operations. Our focus is on two separate NoSQL databases, MongoDB and DynamoDB, which fall into various categories within the NoSQL domain.NoSQL databases can be divided into four categories. (1) Key-Value stores (DynamoDB, Redis, BerkeleyDB); (2) Document databases (MongoDB, DocumentDB, CouchDB, OrientDB); (3) Wide-Column stores (Cassandra, Hypertable); and (4) Graph databases (Neo4J, OrientDB, Infinite Graph).Both MongoDB and Amazon DocumentDB fall into the category of Document databases. They store data in a document format, commonly utilizing BSON (Binary JSON) for MongoDB and JSON for DocumentDB. The document model supports flexible and dynamic schemas, making it compatible with a wide range of applications, particularly those working with semi-structured or unstructured data.

MongoDB, positioned as a Document Database, is representative of the model in which data is contained in flexible "documents." These documents, formatted in formats such as JSON or BSON, are at the heart of MongoDB's storage process. MongoDB, known for its flexibility, breaks down the constraints established by inflexible schemas, allowing for the storage of a wide range of data kinds. It excels at organizing large datasets, particularly those including literal documents such as text files, emails, and XML documents.MongoDB stores data in internal memory, which allows for faster data access. It is not necessary to map application objects to database objects in MongoDB. MongoDB can store any sort of data, allowing users to create an infinite number of fields in a document, allowing it to scale more easily. It has a controlled document size of only 16 MB and only 100 levels of performance nesting for documents. Because of the lack of joins, it also demands a large amount of storage space, resulting in data replication. MongoDB does not support transactions, which is a disadvantage that causes data corruption.

Similar to our MongoDB exploration, **Amazon DocumentDB**, an AWS fully managed document database service, supports MongoDB. In accordance with your suggestion, we begin our trip with Amazon DocumentDB by establishing and configuring the database. We will deviate significantly from the standard and use the AWS Command Line Interface for CRUD tasks, such as creating tables, adding data in JSON format, reading entries, altering specific columns, and executing bulk removals. The thorough measurement and documentation of time required throughout these activities provides a solid platform for a comprehensive performance study, copying the methods employed in the standard.

Amazon DocumentDB is a type of document database. Document databases store, query, and retrieve data in the form of documents, generally utilizing JSON-like structures. DocumentDB is compatible with MongoDB, which implies it supports the document data model and the MongoDB query language. Document databases are suitable for handling semi-structured data and are flexible to modifications in data structure as time progresses.

Key Differences

MongoDB and DocumentDB, while both classified as document databases, have distinct characteristics. DocumentDB stands out when it comes to security, leveraging the advanced AWS Identity and Access Management permission mechanism and adding an additional layer of security by not allowing direct internet access. While MongoDB has excellent security capabilities, it lacks the same level of out-of-the-box security as DocumentDB. Moving on to data size constraints, DocumentDB has a maximum limit of 400 KB, but MongoDB can accept larger data sets with a cap of 16 MB. Furthermore, DocumentDB only supports three data types (number, string, and binary) and offers rudimentary query capability such as key-value inquiries, with potential data persistence. MongoDB, on the other hand, provides a wider range of data types, such as int, long, date, timestamp, geographic, and floating-point among others. Its query language allows developers to thoroughly analyze data, and it maintains excellent data consistency, distinguishing it from DocumentDB.

Project Plan

This project aims to conduct a comprehensive comparison between MongoDB and DocumentDB, utilizing the IMDb movie dataset as an example. We will perform CRUD operations on this dataset, evaluating the differences in time taken by both databases and assessing their overall performance based on the obtained results

Step 1 - Configuration and Setup

We need to first create MongoDB and DocumentDB databases, then configure their environments and establish both a MongoDB cluster and an AWS DocumentDB cluster through the use of the AWS CLI.

Step 2 - Performance Evaluation

Involves performance benchmarking for MongoDB and DocumentDB using the IMDb dataset, analyzing CRUD operations, and implementing data modeling for IMDb dataset-specific use cases.

Step 3 - Use Case Evaluation

The process includes analyzing various use cases in order to uncover the strengths of both MongoDB and DocumentDB, particularly in relation to the IMDb dataset. This analytical approach aims to fully engage and understand the capabilities of these two databases.

Step 4 - CRUD Operations Comparison

Exploring CRUD operations in MongoDB and DocumentDB involves analyzing the time, read, update, and delete processes in order to gauge insertion speed and overall performance of each dataset.

Step 5 - Scalability Assessment

Involves assessing the scalability of MongoDB and DocumentDB, evaluating their performance under dynamic operating scenarios for handling data growth.

Step 6 - Evaluation of Real-Time Responsiveness

We will conduct a live analysis of MongoDB and DocumentDB using the IMDb dataset, in order to compare their read and write speeds for varying workloads and amounts of data.

Step 7 - Arithmetic Operations

Involves performing arithmetic operations on the IMDb dataset, including determining minimum and maximum values, mean, and standard deviation, time these operations, and compare the results.

Step 8 - Reporting and documentation

The process includes examining performance tests, conducting use case analysis, assessing CRUD operations, measuring scalability, evaluating latency, and performing arithmetic operations. This thorough evaluation results in a detailed report and tailored suggestions for choosing the best database option.

Milestones

TASK	ASSIGNED TO - ETA
Data setup - MongoDB	Sailavanya Narthu - EOD
Data setup - DocumentDB	Arundhathi Nalla - EOD
CRUD - MongoDB	Sai Sandeep Neerukonda - EOD
CRUD - DocumentDB	Arundhathi Nalla - EOD
Performance comparison	Sai Sandeep Neerukonda - EOD
Final Report	Sailavanya Narthu - EOD

Works Cited:

https://www.guru99.com/nosql-tutorial.html#:~:text=NoSQL%20Databases%20are%20mainly%20categorized,%2Dbased%20and%20Document%2Doriented
https://www.mongodb.com
https://acodez.in/mongodb-nosql-database/
https://www.knowledgenile.com/blogs/pros-and-cons-of-mongodb
https://medium.com/analytics-vidhya/crud-operations-in-mongodb-using-python-49b7850d627e
https://hevodata.com/learn/mongodb-crud-operations-in-python/
https://news.ycombinator.com/item?id=18869755
https://developer.imdb.com/non-commercial-datasets/ imdb movie dataset link
https://aws.amazon.com/blogs/database/introduction-to-data-modeling-with-amazon-documentdb-with-mongodb-compatibility-for-relational-database-users/
https://www.mongodb.com/compare/documentdb-vs-mongodb