|  |  |  |
| --- | --- | --- |
| **UMBC – CSEE Department** |  | Dr. Waleed Youssef |
| **Data Science Program** |  | youssef1@umbc.edu |
| **Fall 2023** |  |  |
| **DATA 603 – Big Data Platforms** | | |
| **Homework #4** | | |
| Name: Akshay Reddy Gone  Id: AM60898 |  |  |

**Questions:**

1. **[10 points]** Describe when NoSQL database is preferred to RDBMS? Present benefits and different scenarios.

In a variety of scenarios, NoSQL RDBMS are preferred over relational database management systems in view of the specific advantages that they offer. The various scenarios and the added advantages of NoSQL databases compared to RDBMS are shown below:

**Dynamic Schema:**

**Scenario:** NoSQL databases are preferred when your application's data structure is ill-defined or dynamic. They are appropriate for applications with changing requirements because they let you add fields without affecting already-existing data.

**Benefits:** NoSQL databases offer adaptability to changing data models without the need for schema migrations.

**Scalability:**

**Scenario:** NoSQL databases excel at scalability when your application experiences rapid growth and needs to handle large volumes of data and high traffic loads.

**Benefits:** NoSQL databases support horizontal scaling, distributing data across multiple servers or nodes, allowing your application to grow without interruption.

**Analytics in real-time and big data:**

**Scenario:** In applications involving big data and real-time analytics, where quick data processing and analysis are essential, NoSQL databases are well suited.

**Benefits:** NoSQL databases make it possible to handle large datasets efficiently and support real-time analytics, giving businesses insightful data.

**Nested or Hierarchical Data:**

**Scenario:** NoSQL databases are advantageous for applications that use complex, nested data structures (like JSON or XML), as they can efficiently store and query such data.

**Benefits:** NoSQL databases make it simple to work with hierarchical data, which is typical in situations like IoT sensor data and e-commerce product catalogues.

**High Throughput Writing:**

**Scenario:** NoSQL databases are advantageous for applications that need quick data ingestion and the capacity to handle high write throughput.

**Benefits:** NoSQL databases, in particular column-family and document-oriented databases, are suitable for real-time dashboards and logging systems because they can effectively handle high write loads.

**High Availability and Tolerance to Faults:**

**Scenario:** NoSQL databases are used by critical applications that require high availability and fault tolerance, like online banking or healthcare systems.

**Benefits:** Data replication across numerous servers or data centres is supported by NoSQL databases, ensuring uninterrupted service even in the event of hardware failures.

**Complex Queries:**

**Scenario:** While NoSQL databases can manage complex queries, they might not support SQL-like joins and transactions. However, NoSQL databases are a good option for applications that mainly involve straightforward read and write operations.

**Benefits:** NoSQL databases can execute complex queries within their data model, producing effective results for particular use cases.

**Cost-effective Scaling:**

**Scenario:** NoSQL databases frequently offer a cost-effective way to scale for businesses on a tight budget or for startups.

**Benefits:** NoSQL databases are appealing to businesses with limited resources because horizontal scaling, which involves adding more commodity servers, can be more affordable than vertical scaling a single server.

**High throughput with little lag:**

**Scenario:** The use of NoSQL databases is advantageous for applications that require quick responses and high throughput, such as real-time chat, gaming, and ad tech platforms.

**Benefits:** NoSQL databases provide quick data processing and access, resulting in low-latency responses.

**Benefits:** Low-latency responses are ensured by the quick data access and processing capabilities provided by NoSQL databases.

**Multiple Data Types:**

**Situation:** Applications using a variety of data types, such as text, geospatial, time series, or binary data, find NoSQL databases to be a useful option.

**Benefits:** NoSQL databases can store a variety of data types and are appropriate for geolocation-based services, multimedia content management, and other applications.

In conclusion, because of its flexibility, scale and efficient management of specific types of data, the NoSQL database is preferred over other databases when working with dynamic, unstructured or largescale data. It will depend upon the application's specific needs, type of data and its scalability requirements when you select a NoSQL or RDBMS. Polyglot persistence could be an excellent option for different aspects of your application, including the combination of NoSQL and RDBMS.

1. **[10 points]** Compare MongoDB to Cassandra DB, show differences and discuss advantages and disadvantages?

Both Cassandra and MongoDB are widely used NoSQL databases, but they have different designs, use cases, and benefits and drawbacks. MongoDB and Cassandra are contrasted here:

**Data Model:**

**MongoDB:** A document-based NoSQL database is MongoDB. It organises data into collections of documents and stores it in semi-structured BSON (Binary JSON) format. The structure of each document can vary.

**Cassandra:** Wide-column NoSQL Cassandra is a database. Data is kept in a tabular format with rows and columns. It is geared towards write-intensive workloads.

**Query Language:**

**MongoDB:** Ad-hoc queries can be supported by MongoDB's rich query language, which also supports complex queries. The flexibility of data querying is greater.

**Cassandra:** CQL (Cassandra Query Language), which is similar to SQL but does not support complex queries, is used by Cassandra. For straightforward read and write operations, it is better.

**Scalability:**

**MongoDB:** In order to handle large data volumes and high traffic loads, MongoDB supports horizontal scaling through sharding.

**Cassandra:** Cassandra was created with horizontal scalability in mind and is naturally very distributed. It is simple to scale across various nodes or data centers.

**Continuity Model:**

**MongoDB:** Strong consistency is offered by default in MongoDB, but eventual consistency can be configured.

**Cassandra:** With Cassandra's tunable consistency, you can select between strong and eventual consistency depending on your requirements.

**Use Cases:**

**MongoDB:** Content management systems, catalogues, and blogging platforms are examples of use cases where MongoDB is appropriate. These use cases call for flexibility and complex querying.

**Cassandra:** For use cases requiring high write throughput and scalability, such as time-series data, sensor data, recommendation engines, and other applications, Cassandra is a good option.

**MongoDB's benefits include:**

* For flexible data retrieval, a rich query language is used.
* the ability to scale horizontally through sharding.
* dependable consistency support.

**Use Cases:**

**MongoDB:** Content management systems, catalogues, and blogging platforms are examples of use cases where MongoDB is appropriate. These use cases call for flexibility and complex querying.

**Cassandra:** For use cases requiring high write throughput and scalability, such as time-series data, sensor data, recommendation engines, and other applications, Cassandra is a good option.

**MongoDB's benefits include:**

* For flexible data retrieval, a rich query language is used.
* the ability to scale horizontally through sharding.
* dependable consistency support.

**MongoDB drawbacks:**

* restricted assistance for complicated transactions.
* For workloads with a lot of writing, inefficient.

**Cassandra's benefits include:**

* high scalability and throughput for writes.
* consistent with a range of use cases.
* high availability and fault tolerance.

**Cassandra's drawbacks:**

* Complex query support is not very good.
* Not appropriate for use cases requiring sophisticated transactions.

In conclusion, there are distinct use cases and trade-offs for MongoDB and Cassandra. While Cassandra excels in high write throughput, scalability, and fault tolerance, MongoDB is better suited for applications with complex querying and moderate write loads. The decision between them is based on the needs of your application.

1. **[15 points]** Describe new features of MongoDB not shared in class, especially features available in releases that were not part of older releases.

Several new features and enhancements that weren't present in earlier versions of MongoDB have been added in more recent releases. Some notable features that MongoDB now provides include the ones listed below:

**Client-Side Field-Level Encryption (CSFLE):**

With CSFLE, you can encrypt sensitive fields in your documents before they are stored in the database. CSFLE was introduced by MongoDB in version 4.2. Sensitive data is protected because encryption and decryption take place on the client side, adding an extra layer of data security.

**Atlas Data Lake:**

The managed database service MongoDB Atlas now has a feature called Atlas Data Lake that enables you to directly query and analyze data stored in your AWS S3 buckets. This feature makes it simpler to work with data stored in cloud object storage and simplifies data analysis.

**Change Streams:**

Introduced in version 3.6, Change Streams have undergone significant improvements in later iterations. They give programmes the ability to monitor changes in the database that are occurring in real time. This feature, which starts triggering operations based on database changes, is crucial for creating reactive applications.

**Wildcard Indexes:**

The introduction of wildcard indexes in MongoDB 4.2 made it possible to index fields inside embedded arrays. When dealing with data that has deeply structured or nested arrays, this improves query performance.

**On-Demand Materialised Views:**

As of MongoDB version 4.2, you can build on-demand materialised views to pre-aggregate and denormalize data for reporting and analytical needs. Complex data transformations are made easier by this feature.

**MongoDB Realm:**

Despite not being a component of the database itself, MongoDB Realm is a noteworthy addition. It's a serverless development environment that supports user authentication, data synchronisation, and serverless operations for your MongoDB data, allowing you to build cutting-edge mobile, web, and mobile applications.

**Distributed Transactions:**

In version 4.0, MongoDB added support for multi-document transactions, enabling you to execute several operations concurrently. In complicated operations, this feature guarantees data integrity and consistency.

**Schema Validation:**

Schema validation rules, introduced in MongoDB 3.6, let you specify guidelines for the organisation and content of your documents. This guarantees that your data complies with predetermined standards.

**Serverless Instances:**

With the introduction of serverless instances in MongoDB Atlas, you can now automatically scale the compute capacity of your database up or down in response to workload demands, saving money during periods of low activity.

**MongoDB Charts:**

Introduced as a data visualisation tool that integrates seamlessly with MongoDB data, despite not being a component of the database itself. You can make and distribute interactive charts and dashboards using this tool.

Recent updates to MongoDB have increased its functionality and made it more adaptable and user-friendly for a variety of use cases. Among other things, they address requirements for data analysis, real-time data streaming, and data security.

1. **[25 points]** MongoDB Implementation Assignment:
2. Create and develop a new MongoDB database for car makes and models.
3. Generate one report of data loaded, showing different commands applied on this database (load at least 100 records in the database).
4. Generate one report showing how many models you entered per car make.
5. Generate another report showing how many American-made cars versus rest of the world.
   * ***Hints:***
     + *In your report, you should list all commends used starting from creating the database, creating collections and documents, inserting data, and querying/filtering data.*
     + *Show your work to get full credit!*

**Attached report document for this question.**

**References:**

1. **Installation :** [**https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-os-x/**](https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-os-x/)