DAY-7(NoSQL)

1. NoSQL Databases:
   a. Write a Python program that connects to a MongoDB database and inserts a new document into a collection named "students". The document should include fields such as "name", "age", and "grade". Print a success message after the insertion.
   b. Implement a Python function that connects to a Cassandra database and inserts a new record into a table named "products". The record should contain fields like "id", "name", and "price". Handle any potential errors that may occur during the insertion.

2. Document-oriented NoSQL Databases:
   a. Given a MongoDB collection named "books", write a Python function that fetches all the books published in the last year and prints their titles and authors.
   b. Design a schema for a document-oriented NoSQL database to store customer information for an e-commerce platform. Write a Python program to insert a new customer document into the database and handle any necessary validations.

3. High Availability and Fault Tolerance:
   a. Explain the concept of replica sets in MongoDB. Write a Python program that connects to a MongoDB replica set and retrieves the status of the primary and secondary nodes.
   b. Describe how Cassandra ensures high availability and fault tolerance in a distributed database system. Write a Python program that connects to a Cassandra cluster and fetches the status of the nodes.

4. Sharding in MongoDB:
   a. Explain the concept of sharding in MongoDB and how it improves performance and scalability. Write a Python program that sets up sharding for a MongoDB cluster and inserts multiple documents into a sharded collection.
   b. Design a sharding strategy for a social media application where user data needs to be distributed across multiple shards. Write a Python program to demonstrate how data is distributed and retrieved from the sharded cluster.

5. Indexing in MongoDB:
   a. Describe the concept of indexing in MongoDB and its importance in query optimization. Write a Python program that creates an index on a specific field in a MongoDB collection and executes a query using that index.
   b. Given a MongoDB collection named "products", write a Python function that searches for products with a specific keyword in the name or description. Optimize the query by adding appropriate indexes.

Submission Guidelines:

- Answer all the questions in a single Jupyter Notebook file (.ipynb).
- Include necessary code, comments, and explanations to support your answers and implementation.

- Ensure the notebook runs without errors and is well-organized.
- Create a GitHub repository to host your assignment files.
- Rename the Jupyter Notebook file using the format "date_month_topic.ipynb" (e.g., "12_July_NoSQL.ipynb").
- Place the Jupyter Notebook file in the repository.
- Commit and push any additional files or resources required to run your code (if applicable) to the repository.
- Ensure the repository is publicly accessible.
- Submit the link to your GitHub repository as the assignment submission.

Grading Criteria:

1. Understanding and completeness of answers: 40%
2. Clarity and depth of explanations: 25%
3. Correct implementation and evaluation of optimizer techniques: 15%
4. Analysis and comparison of different optimizers: 10%
5. Proper code implementation and organization: 10%

Note:- Create your assignment in Jupyter notebook and upload it to GitHub & share that uploaded assignment file link through your dashboard. Make sure the repository is public.