Ira A. Fulton Schools of Engineering
Arizona State University

Term: Fall 2023 **Subject:** Computer Science & Engineering (CSE) **Number:** 512
Course Title: **D**istributed **D**atabase **S**ystems (CSE 512)

Team: PVS

Part 5: Distributed NoSQL Database Systems Implementation

Problem Statement: Implement a distributed NoSQL database system to understand data storage and retrieval process with your chosen domain.

Code:

```
CREATE KEYSPACE stock_data WITH replication = {'class': 'SimpleStrategy',
'replication_factor': 3};


USE stock_data;

CREATE TABLE stock_prices (
   symbol TEXT,
   date DATE,
   price DECIMAL,
   PRIMARY KEY (symbol, date)
) WITH CLUSTERING ORDER BY (date DESC);


INSERT INTO stock_prices (symbol, date, price) VALUES ('AAPL', '2023-01-01', 150.25);
INSERT INTO stock_prices (symbol, date, price) VALUES ('AAPL', '2023-01-02', 152.50);


SELECT * FROM stock_prices WHERE symbol = 'AAPL' AND date >= '2023-01-01' AND
date <= '2023-01-02';
```

2. Implement basic CRUD (Create, Read, Update, Delete) operations for the domain-specific data.

Code:

```
from pymongo import MongoClient
from datetime import datetime

# Connect to MongoDB
```

```
client = MongoClient('mongodb://localhost:27017/')
db = client['stock_data']
collection = db['stock_prices']

# Create
def create_stock_price(symbol, date, price):
    data = {
        'symbol': symbol,
        'date': date,
        'price': price
    }
    result = collection.insert_one(data)
    print(f"Inserted document with _id: {result.inserted_id}")

# Read
def read_stock_prices(symbol, start_date, end_date):
    query = {
        'symbol': symbol,
        'date': {'$gte': start_date, '$lte': end_date}
    }
    cursor = collection.find(query)
    for document in cursor:
        print(document)

# Update
def update_stock_price(symbol, date, new_price):
    query = {'symbol': symbol, 'date': date}
    new_values = {'$set': {'price': new_price}}
    result = collection.update_one(query, new_values)
    print(f"Modified {result.modified_count} document")

# Delete
def delete_stock_price(symbol, date):
    query = {'symbol': symbol, 'date': date}
    result = collection.delete_one(query)
    print(f"Deleted {result.deleted_count} document")

# Example Usage
create_stock_price('AAPL', datetime(2023, 1, 1), 150.25)
create_stock_price('AAPL', datetime(2023, 1, 2), 152.50)

read_stock_prices('AAPL', datetime(2023, 1, 1), datetime(2023, 1, 2))

update_stock_price('AAPL', datetime(2023, 1, 1), 155.75)

delete_stock_price('AAPL', datetime(2023, 1, 2))
```

```
# Close MongoDB connection
client.close()
```

3. Create sample queries and data retrieval operations to
showcase the functionality of your NoSQL database for your
chosen topic.
Deliverables Code/Script, Snapshots, Documentation
Possible
Tools
MongoDB, Cassandra, Redis, Couchbase, (or) any other of your
choice.

Code:

Below is an example Python script using the pymongo library to implement CRUD operations for stock pricing historical data in MongoDB. This example assumes that you have a running MongoDB instance locally.

```
from pymongo import MongoClient
from datetime import datetime

# MongoDB connection setup
client = MongoClient('localhost', 27017)
db = client['stock_data']
collection = db['stock_prices']

# Sample Data Insertion
def insert_sample_data():
    create_stock_price('AAPL', datetime.strptime('2023-01-01', '%Y-%m-%d'), 150.25)
    create_stock_price('AAPL', datetime.strptime('2023-01-02', '%Y-%m-%d'), 152.50)
    create_stock_price('GOOGL', datetime.strptime('2023-01-01', '%Y-%m-%d'), 2000.75)
    create_stock_price('GOOGL', datetime.strptime('2023-01-02', '%Y-%m-%d'), 2050.50)

# Create Operation
def create_stock_price(symbol, date, price):
    document = {
        'symbol': symbol,
        'date': date,
        'price': price
    }
    collection.insert_one(document)

# Read Operation
def read_stock_prices(symbol, start_date, end_date):
    query = {
        'symbol': symbol,
        'date': {'$gte': start_date, '$lte': end_date}
    }
    result = collection.find(query)
    return result

# Update Operation
def update_stock_price(symbol, date, new_price):
    query = {
        'symbol': symbol,
        'date': date
    }
    update_query = {
        '$set': {'price': new_price}
    }
    collection.update_one(query, update_query)

# Delete Operation
def delete_stock_price(symbol, date):
    query = {
        'symbol': symbol,
        'date': date
    }
    collection.delete_one(query)

# Sample Queries
def showcase_functionality():
    print("Sample Query 1 - Retrieve Stock Prices for AAPL from 2023-01-01 to 2023-01-02:")
    print(list(read_stock_prices('AAPL', '2023-01-01', '2023-01-02')))

    print("\nSample Query 2 - Retrieve Stock Prices for GOOGL from 2023-01-01 to 2023-01-
02:")
    print(list(read_stock_prices('GOOGL', '2023-01-01', '2023-01-02')))

    print("\nSample Query 3 - Retrieve All Stock Prices for AAPL:")
    print(list(read_stock_prices('AAPL', '1970-01-01', '2100-01-01')))

# Execute Sample Data Insertion
insert_sample_data()

# Execute Sample Queries
showcase_functionality()

# Close MongoDB connection
client.close()
```