Sinem Arslan ER-1690

Cassandra for Library Management

Introduction

The project aims to implement a distributed library management system using Cassandra, with primary functionalities including adding new books, updating book information, adding new users, reservations, and more. Cassandra was selected based on the teacher's recommendation, while my personal interest in books influenced the choice of a library system.

System Description

The system is built on a 4-node Cassandra cluster, using Docker containers for ease of setup and management. The application includes features for making, updating, and viewing reservations, all of which must be operational across all nodes in the cluster.

Implementation Details

Docker Setup

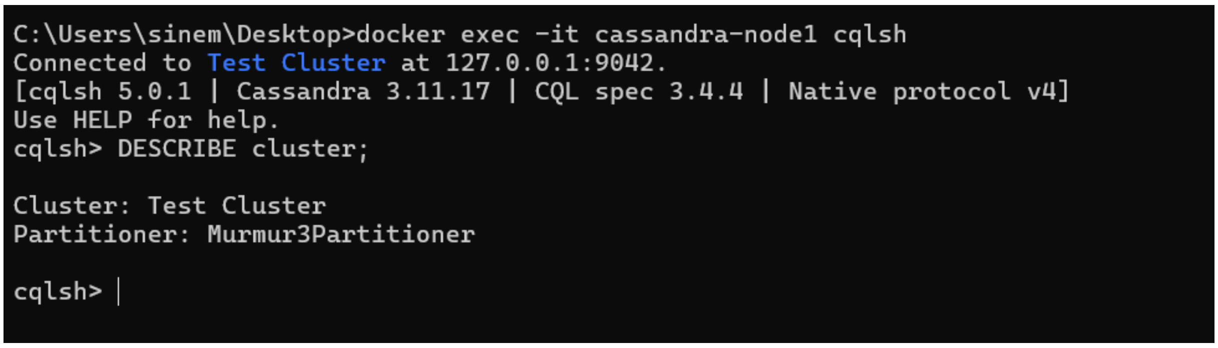
```
docker network create cassandra-network

docker run --name cassandra-node1 --network cassandra-network -d cassandra:3.11

docker run --name cassandra-node2 --network cassandra-network -e
CASSANDRA_SEEDS=cassandra-node1 -d cassandra:3.11

docker run --name cassandra-node3 --network cassandra-network -e
CASSANDRA_SEEDS=cassandra-node1 -d cassandra:3.11

docker run --name cassandra-node4 --network cassandra-network -e
CASSANDRA_SEEDS=cassandra-node1 -d cassandra:3.11
```



```
C:\Users\sinem\Desktop>docker exec -it cassandra-node1 cqlsh
Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.17 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh> DESCRIBE cluster;

Cluster: Test Cluster
Partitioner: Murmur3Partitioner

cqlsh> |
```

Database Setup

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

USE library;

CREATE TABLE books (
  book_id UUID PRIMARY KEY,
  title TEXT,
  author TEXT,
  published_date DATE,
  genre TEXT,
  available BOOLEAN
);

CREATE TABLE reservations (
  reservation_id UUID PRIMARY KEY,
  book_id UUID,
  user_id UUID,
  reservation_date TIMESTAMP,
  status TEXT
);

CREATE TABLE users (
  user_id UUID PRIMARY KEY,
  name TEXT,
  email TEXT,
  join_date DATE
);
```

Database Schema

Keyspace: library

Replication: SimpleStrategy with replication factor 3

The schema designed for this project includes three main tables:

1. books: Stores information about each book available in the library.
   - ‘book_id’ (UUID, Primary Key): Unique identifier for each book.
   - ‘title’ (Text): Title of the book.
   - ‘Author’ (Text): Author of the book.
   - ‘published_date’ (Date): Date of publication.
   - ‘genre’ (Text): Genre of the book.
   - ‘available’ (Boolean): Availability status of the book.reservations:

2. reservations: Manages reservations made by users for specific books.
   - ‘reservation_id’ (UUID, Primary Key): Unique identifier for each reservation.
   - ‘book_id’ (UUID): Foreign key referencing books.book_id.
   - ‘user_id’ (UUID): Identifier of the user making the reservation.
   - ‘reservation_date’ (Timestamp): Date and time when the reservation was made.
   - ‘status’ (Text): Current status of the reservation (e.g., reserved, canceled).

3. users: Contains user information who make reservations.
   - ‘user_id’ (UUID, Primary Key): Unique identifier for each user.
   - ‘name’ (Text): Name of the user.
   - ‘email’ (Text): Email address of the user.
   - ‘join_date’ (Date): Date when the user joined the library.

Problems Encountered

During the implementation of the Cassandra-based library system, several challenges were encountered:

Designing tables and system:

Designing tables to support different types of queries was a important part of the project and quite challenging to me. I needed to create queries for tasks like fetching available books, retrieving reservations by user, and updating reservation statuses.

Stress Testing and Performance:

The system struggled with performance during high-intensity stress tests.

Error Handling:

Managing the entire Cassandra system was difficult for me, and I encountered numerous errors throughout the project. I attempted to resolve these issues through extensive research.