Paseo Posse – Hyperlocal Delivery System

MongoDB is utilized for its NoSQL capabilities, specifically leveraging sharding and replication to address the requirements of scalability, high availability, and fault tolerance.

**Sharding in MongoDB:**

**Horizontal Scaling:**

Sharding in MongoDB enables horizontal scaling by distributing data across multiple servers (shards). This is particularly advantageous for handling the geographical dispersion of data, which is critical in a hyperlocal delivery system where data is partitioned based on ZIP codes.

**Sharding Key:**

The choice of sharding key, in this case, is ZIPCODE. Sharding based on ZIP codes allows the system to distribute data geographically, improving read and write performance by distributing the data load across multiple shards.

**Replication in MongoDB:**

**High Availability:**

MongoDB's replication is employed to achieve high availability and fault tolerance. Each shard in the sharded cluster consists of a three-member PSS (Primary-Secondary-Secondary) replica set. This setup ensures that even if one node fails, there are redundant copies of data on other nodes.

**Fault Tolerance:**

In the master-slave replication setup, one node serves as the master that accepts write operations, while the other nodes (slaves) replicate data from the master and serve read operations. This redundancy enhances fault tolerance and ensures data availability in case of node failures.

**MongoDB Collections:**

**CUSTOMER\_DETAILS Collection:**

Denormalization Technique: Storing the last 4 orders directly in the customer document optimizes read operations for common use cases, such as viewing recent order history. This denormalization is a trade-off between read and write efficiency, aiming to improve performance for frequently accessed data.

**Example entry:**

*{*

*"CUSTOMER\_ID": 1,*

*"CUSTOMER\_NAME": "John Doe",*

*"ZIPCODE": "12345",*

*"ORDER\_INFO": [*

*{"ORDER\_ID": "A123", "STATUS": "Delivered"},*

*{"ORDER\_ID": "B456", "STATUS": "Shipped"},*

*{"ORDER\_ID": "C789", "STATUS": "Processing"},*

*{"ORDER\_ID": "D012", "STATUS": "Delivered"}*

*]*

*}*

**MEDICINE\_DETAILS Collection:**

Stores static information about medicines, including ID, name, and price. This collection is essential for maintaining details about the available medicines in the system.

**Example entry:**

*{*

*"MEDICINE\_ID": 1,*

*"MEDICINE\_NAME": "Aspirin",*

*"PRICE": 5*

*}*

**WAREHOUSE\_DETAILS Collection:**

Stores details about warehouses, including their ID, name, and ZIP code. This collection helps manage the distribution and availability of goods across different warehouses.

**Example entry:**

*{*

*"WAREHOUSE\_ID": 1,*

*"WAREHOUSE\_NAME": "Central Warehouse",*

*"ZIPCODE": "88345"*

*}*

**Replication for High Availability:**

Objective:

MongoDB's replication is used to enhance high availability and fault tolerance. Replication maintains multiple copies of data across different servers, ensuring data availability even in the event of node failures.

Replication Setup:

The MongoDB sharded cluster includes a configuration server replica set and three shards, each comprising a three-member PSS (Primary-Secondary-Secondary) replica set. This architecture provides redundancy and fault tolerance.

Fault Tolerance:

The replication setup ensures that even if a node fails, data remains available, and the system can continue operations without significant disruptions.