Bicol University
Bicol University Polangui
Polangui, Albay

National Bookstore Inventory Management

Information Management

Kyla Shaina Layosa
Lovely Irish Benoyo
Kristina Paula Alamil
Shane Yvette Penaflor
Nicole Anne Bellen
Akisha Miel Reomalis

BSIS-2B

In an Inventory Management System, one key area that can benefit from NoSQL integration is the **Audit Logging** system. While relational databases like MySQL are well-suited for managing structured data (e.g., product details, inventory levels), they can become inefficient when dealing with high-volume, schema-less log entries. **NoSQL databases**, particularly document-based ones like **MongoDB**, provide a more scalable and flexible approach for managing this kind of data.

Why NoSQL for Audit Logs?

Audit logs typically record user actions, system events, and transactional history—data that is often semi-structured and dynamic. Implementing this in a relational model would require constant schema updates, complex indexing, and significant storage overhead. In contrast, MongoDB can store these logs in a JSON-like format, allowing for easy insertion of new types of logs without altering the existing structure.

Storing and Querying Logs with MongoDB

Instead of storing logs in rigid SQL tables, MongoDB enables storage as documents within collections. For example:

```
{
  "user_id": 123,
  "action": "Update",
  "table": "Books",
  "timestamp": "2024-03-
31T12:45:00Z",
  "details": {
    "field": "price",
    "old_value": 19.99,
    "new_value": 24.99
  }
}
```

MongoDB’s dynamic schema allows new fields and data types to be added without migrations. Querying recent activities becomes efficient using its aggregation framework. A typical query might look like:

db.audit_logs.find({ "user_id": 123 }).sort({ "timestamp": -1 }).limit(10)

CONCLUSION

Incorporating NoSQL for audit logging enhances **scalability**, **performance**, and **flexibility**, especially for handling unstructured or semi-structured data in real time. A **hybrid approach**—using MySQL for structured core inventory data and MongoDB for flexible logging—represents an optimized data management solution. This strategy improves system reliability and supports efficient, real-time data processing (Li et al., 2021).