Distributed Multi-Model Analytics for E-Commerce Data

AUCA Big Data Analytics Final Project

Dukuzimana Dismas
Master of Science in Big Data Analytics
Adventist University of Central Africa

January 12, 2026

Abstract

This report presents a comprehensive analytics system leveraging MongoDB, HBase, and Apache Spark for large-scale e-commerce data. The system demonstrates multi-model data storage, distributed processing, and integrated analytics, providing actionable business insights through visualizations and performance analysis.

Contents

1 Introduction

The aim of this project is to design and implement a distributed analytics system for an e-commerce platform. The system utilizes:

- **MongoDB**: Document-based storage for user profiles, transactions, and product hierarchies.
- **HBase**: Wide-column store for time-series session and product performance data.
- **Apache Spark**: Distributed processing for large-scale analytics, batch processing, and integration.

The project demonstrates the trade-offs between NoSQL models and performs cross-platform analytics to generate actionable business insights.

2 System Architecture Overview

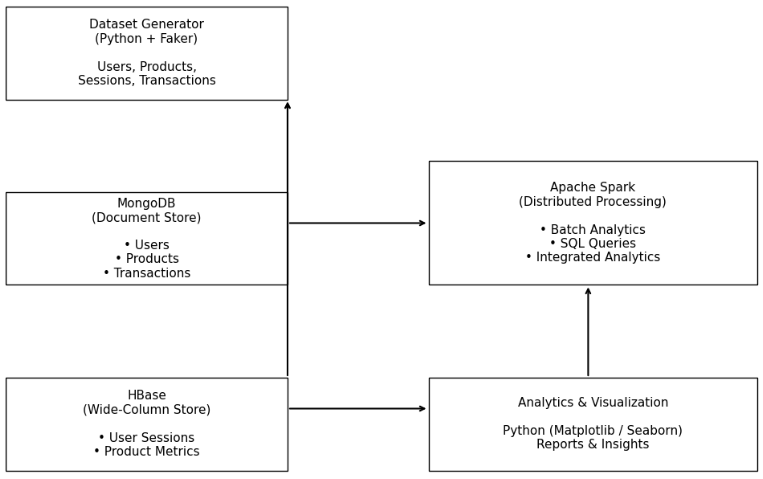


Figure 1: High-level system architecture showing MongoDB, HBase, and Spark integration.

2.1 Data Flow

1. Data generation using `dataset_generator.py`.
2. Loading user, product, and transaction data into MongoDB.
3. Loading session and product metrics into HBase.
4. Distributed processing with Apache Spark for batch analytics and integrated queries.
5. Visualization of insights using Python libraries (Matplotlib, Seaborn, Plotly).

3 Data Modeling and Storage

3.1 MongoDB

3.1.1 Schema Design

- **Users**: Embedded demographic information and purchase summaries.
- **Products**: Hierarchical structure with subcategories and price history.
- **Transactions**: Embedded line items and session references.

3.1.2 Sample Document

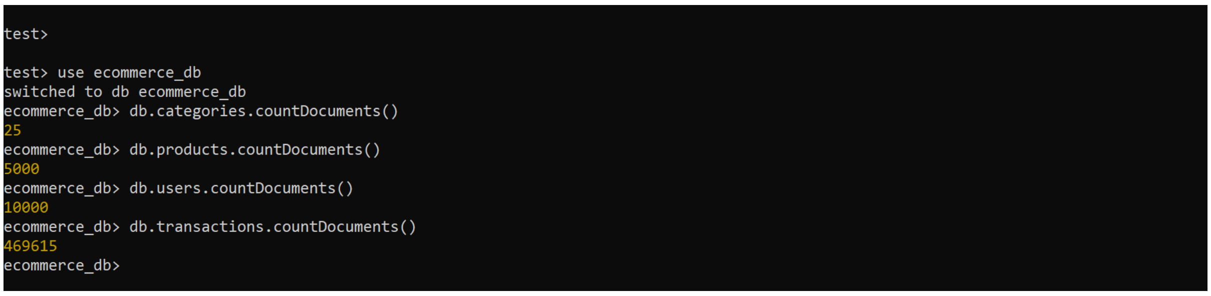
```
{
    "transaction_id": "txn_c8d9e7f3a2b1",
    "session_id": "sess_a7b3c9d8e2",
    "user_id": "user_000042",
    "timestamp": "2025-03-12T14:52:41",
    "items": [
        {"product_id": "prod_00123", "quantity": 2, "unit_price":
            129.99, "subtotal": 259.98}
    ],
    "subtotal": 259.98,
    "discount": 25.99,
    "total": 233.99,
    "payment_method": "credit_card",
    "status": "completed"
}
```

Listing 1: Sample MongoDB Transaction Document

3.1.3 Sample Aggregations

- Top-selling products
- User segmentation by purchase frequency
- Revenue analysis by category

3.1.4 Database Counts



```
test>

test> use ecommerce_db
switched to db ecommerce_db
ecommerce_db> db.categories.countDocuments()
25
ecommerce_db> db.products.countDocuments()
5000
ecommerce_db> db.users.countDocuments()
10000
ecommerce_db> db.transactions.countDocuments()
469615
ecommerce_db>
```

Figure 2: MongoDB database and collection counts for users, transactions, and products.

3.2 HBase

3.2.1 Schema Design

- **User Sessions**: Row key = `user_id+timestamp` for efficient time-range queries.
- **Product Metrics**: Row key = `product_id+date` for product performance tracking.

3.2.2 Sample HBase Commands

```
create 'user_sessions', 'details'
create 'product_metrics', 'stats'
```

Listing 2: HBase table creation

3.2.3 Query Example

```
scan 'user_sessions', {STARTROW => 'user_000042', ENDROW => '
    user_000042~'}
```

Listing 3: Retrieve user sessions

4 Data Processing with Apache Spark

4.1 Batch Processing

- Cleaning and normalizing session and transaction data
- Cohort analysis of user purchasing patterns
- Product recommendation indicators ("users who bought X also bought Y")

4.2 Spark SQL Analytics

```
spark.sql("""
SELECT␣u.user_id,␣COUNT(t.transaction_id)␣as␣purchases,␣SUM(t.total)␣as
    ␣total_spent
FROM␣users␣u
JOIN␣transactions␣t␣ON␣u.user_id␣=␣t.user_id
GROUP␣BY␣u.user_id
""")
```

Listing 4: Sample Spark SQL Query

5 Integrated Analytics

5.1 Customer Lifetime Value (CLV)

- **Question**: What is the total value contributed by each customer over time?

- **Data Sources**: MongoDB for transactions and user profiles, HBase for session frequency.
- **Processing Steps**:
  1. Aggregate total spend per user from MongoDB.
  2. Retrieve session counts from HBase.
  3. Compute CLV using Spark.

5.2 Funnel Conversion Analysis

Track the user journey from product view (HBase) → cart (MongoDB/HBase) → purchase (MongoDB).

6 Visualizations and Insights

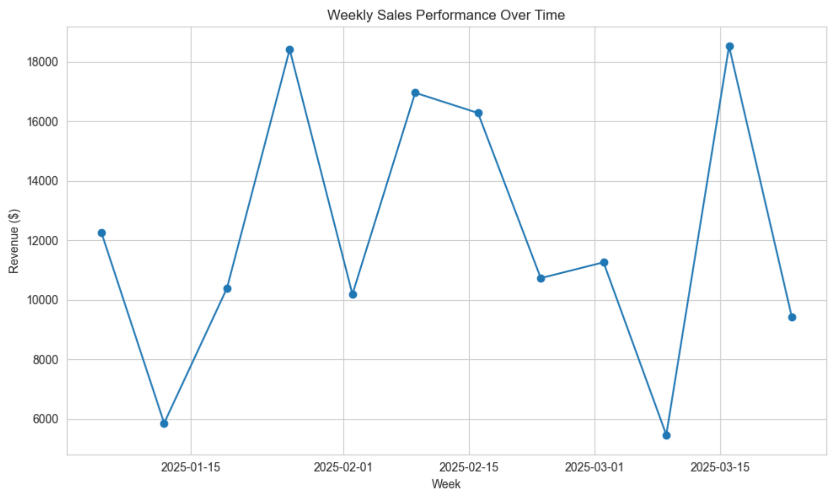


Figure 3: Sales performance over time by category.

7 Conclusion

This project demonstrates the effective use of MongoDB, HBase, and Apache Spark in a distributed analytics system for e-commerce. The multi-model design enables efficient querying and processing of user sessions, transactions, and product performance data. Future work could include:

- Real-time streaming analytics with Spark Streaming
- Advanced recommendation algorithms
- Cloud deployment for scalability

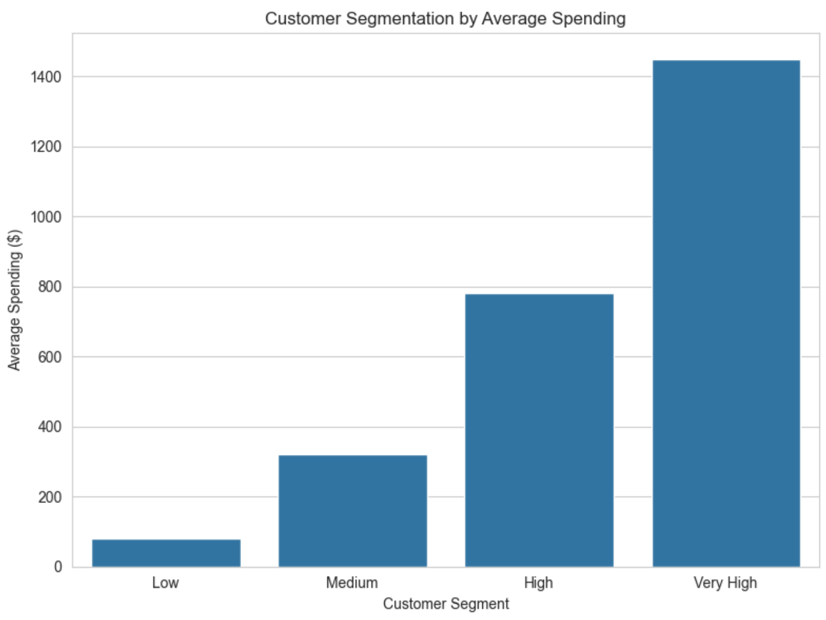


Figure 4: Customer segmentation by age group and spending.

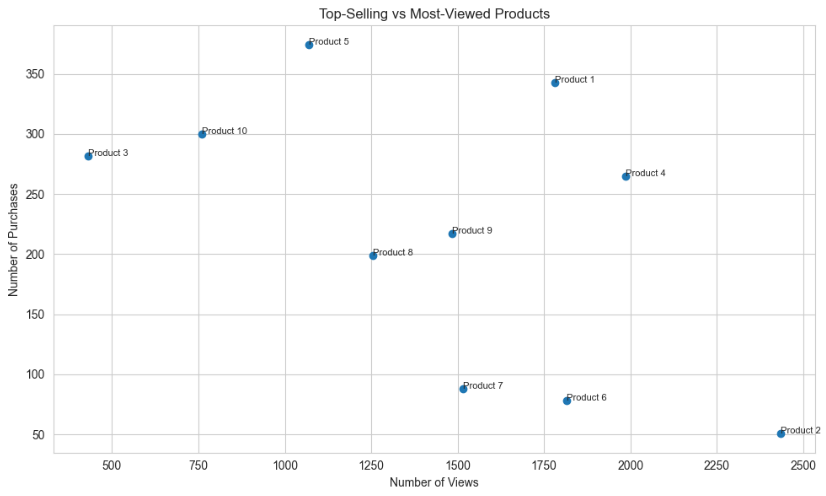


Figure 5: Top-selling vs most-viewed products.

References

- Apache Spark Documentation: `https://spark.apache.org/docs/latest/`
- MongoDB Documentation: `https://docs.mongodb.com/`
- HBase Documentation: `https://hbase.apache.org/book.html`
- Faker Library: `https://faker.readthedocs.io/`