Advanced Databases

2025/2026

Profs. Márcia Barros and Francisco M. Couto

Project description: Recommendation System

Overview

This project focuses on the design, implementation, and optimization of a **recommendation system** that integrates a **relational database (MySQL)** and a **NoSQL document database (MongoDB)**.
Students will design schemas, implement queries, test for concurrency, and optimize both systems so they **complement each other** in supporting recommendation functionality.

Project Description

You will build a recommendation system that uses **structured data stored in MySQL** (e.g., users, transactions, or item metadata) and **semi-structured data stored in MongoDB** (e.g., user preferences, item details, or interaction logs).
The two databases should **interoperate** - data stored in one should enhance queries and functionality in the other.

You are encouraged to choose a domain of personal or academic interest (e.g., courses, books, movies, products, or restaurants).

Infrastructure

- **Relational Database:** MySQL
- **NoSQL Document Database:** MongoDB
- Python and the libraries used in class (pymongo, mysql.connector, sqlalchemy, etc)

Data Source

- Select a dataset from **Kaggle**.
- The dataset must:
    - Be suitable for creating both relational (with more than two tables) and document-oriented structures.
    - Must have the columns referents to user, item and rating.
    - At least 50k ratings.

Example datasets:

- RetailRocket recommender system dataset
- MovieLens 100k
- Book Recommendation Dataset

Project Goals and Tasks

Goal 1 – Data Selection, Modeling, and Database Creation

- Select and analyze the dataset to identify relationships and data hierarchies.
- Design **a relational schema** for MySQL and **a document schema** for MongoDB.
- Decide which data should reside in each system:
    - **MySQL:** structured, transactional, or reference data.
    - **MongoDB:** nested, user-driven, or flexible data.
- Clean (if needed) and import data.
- Ensure integration points between the databases (e.g., shared user or item IDs).

Deliverables (to show in class):

- ER diagram and schema definitions (SQL DDL + MongoDB schema documentation)
- Data loading scripts

Goal 2 – Query Design and Implementation

- Design **queries and operations** that power a basic recommendation system (must include simple and complex queries).
- MySQL should handle structured analytics (e.g., top-rated items, user history).
- MongoDB should handle contextual or preference-based queries (e.g., user interests, item attributes).
- Implement at least **one combined or federated operation** where data from both systems contribute to a recommendation (e.g., fetching user data from MySQL and preference data from MongoDB).

Deliverables (to show in class):

- SQL and MongoDB queries with documentation
- Examples of query outputs
- Explanation of how each query supports recommendations

Goal 3 – Concurrency Testing

- Develop scripts or programs that **simulate multiple concurrent users** accessing and updating both databases.
- Measure:

    - Response times
    - Conflicts or deadlocks
    - Consistency of cross-database operations
- Document your testing setup (tools, concurrency level, sample code).

Deliverables (to show in class):

- Concurrency test scripts
- Performance metrics
- Summary of concurrency behavior and findings

Goal 4 – Performance Optimization and Integration Tuning

- Apply **indexing, query optimization, and schema tuning** in both MySQL and MongoDB.
- Improve **data access patterns** between the two systems (e.g., batching reads/writes, minimizing redundant joins).
- Re-test query performance after optimization.
- Reflect on the **benefits and trade-offs** of the hybrid approach.

Deliverables (to show in class):

- Table with Before-and-after time performance comparison
- Explanation of optimization techniques used
- Final integrated system diagram

Final Delivery:

Date: December 7th, 2025 (23:59)

Moodle

Zip file with code and report

Zip file name: BDA2526_GroupNumber.zip, example BDA2526_G01.zip

The report:

- Maximum of 8 (eight) pages and 1500 words
- Organized by goals
- Discussion of points done/not done in the project
- Description of how to replicate the project: creation of the databases, and running the queries
- All group members should be identified in the report

AI Usage Policy for Student Projects

- Students may use AI tools and platforms (such as ChatGPT, Copilot, Gemini, etc.) to assist with their projects. However, all students are fully responsible for understanding and being able to explain every part of the code, text, or information included in their work.
- AI tools may not be used to generate an entire project or major sections of it. If a project appears to be fully AI-generated or if a student cannot explain their own code or content, the project will receive a grade of zero.

Students must:

- Be able to demonstrate and explain how their code or work functions.