Workload-Aware Tuple Partitioning in Distributed Database Systems

This project evaluates multiple tuple partitioning algorithms for large-scale distributed database systems using the Amazon Book Reviews dataset. The goal is to show that different workloads benefit from different data partitioning strategies, and no single strategy performs optimally for all query patterns.

This work corresponds to Assignment 2 in the course Distributed Database Systems.

Problem Overview

Distributed databases partition data across multiple nodes. However, applications generate workloads with different access patterns, such as:

- User-centric lookups
- Analytical range scans
- Recommendation / co-access patterns

A single partitioning strategy cannot efficiently handle all these workloads.

Objective:

Evaluate four partitioning strategies under three workload types and compare latency, number of partitions accessed, and load balance.

Dataset

Amazon Book Ratings Dataset

After preprocessing:

996,530 interactions

80,699 users

32,930 books

Preprocessing Steps:

- Removed duplicates
- Kept latest review per (user, book)
- Mapped IDs to indices
- Removed inactive users/books (<5 interactions)
- Computed popularity for range partitioning

Partitioning Strategies (P=4)

1. Round Robin - Balanced baseline

2. Hash(User) - Best for lookup workloads

3. Range(Popularity) - Best for scan analytics

4. Group / Co-Access - Best for recommendation workloads

Workloads

W1: User Lookup

W2: Range Scan

W3: Co-Access Bundles

Simulation Model

latency_ms = 0.05 * (rows_scanned / 1000) + 2.0 * (partitions_touched - 1)

Key Findings

W1 → Hash(User)

W2 → Range(Popularity)

W3 → Group/Co-Access

Conclusion:

No single partitioning method is universally optimal. Partitioning must be workload-aware.

Group 7 — Distributed Database Systems, IIT Jodhpur