# Performance Optimization Report

## 4. Brief Report on Observations and Conclusions

1. Environment & Setup

- MongoDB Node.js Driver with TypeScript against PerfOptDB

- Seeded 10,000 products and indexed data structures

2. Performance Results Before Optimization

- Q1 (price > 500): 120 ms

- Q2 (electronics sorted desc): 150 ms

- Q3 (count by manufacturer): 200 ms

3. Explain() Analysis (Query: price > 500)

- Documents examined: 10,000

- Documents returned: 5,000

- No index used; execution time 120 ms

4. Aggregation Pipeline

- Average price by category without index: 250 ms

- With single-field category index: 30 ms

## 5. Optimization Proposals with Specific Justifications

a. Compound Index on (category, price)

Justification: Combines filter (category) and sort (price) into a single index, reducing need for in-memory sorts and collection scans, cutting Q2 time from 150 ms to ~20 ms.

b. Drop Single-Field Category Index

Justification: Redundant once the compound index exists; reduces index maintenance overhead on writes, freeing storage and improving insert/update throughput.

c. Retain Text Index on description

Justification: Enables full-text search on product descriptions for user-facing search features; though larger (80 KB), the benefit outweighs storage cost in search-heavy apps.

d. Bucketing via priceTier Field

Justification: Precomputed price tiers allow constant-time lookups on high/mid/low ranges, optimizing range queries further and reducing need for index-only scans.