Redis–MongoDB Data Pipeline Architecture

Data Sources → ETL → Redis Cache → MongoDB Atlas → Reporting/Dashboard

Flow:

- Data collected from IoT/API sensors
- ETL transforms data to JSON format
- Redis stores frequently accessed readings
- MongoDB stores long-term sensor history
- Cache refresh & TTL maintain data freshness

Lessons Learned

- Redis caching significantly reduced latency (~100 ms → ~5 ms)

- TTL & Cache-Aside strategies balance performance and freshness

- Integration of Redis + MongoDB improved scalability

- Logging & benchmarking helped query optimization

- Cache invalidation policies prevent stale data

Challenges & Outcomes

What worked well:

- Stable Redis connection

- Fast reads under high load

- Streamlined ETL flow into MongoDB

What didn’t work well:

- Cache eviction & TTL tuning needed trial & error

- Occasional MongoDB delays on cache reload

- Concurrent updates required additional logic

Outcome:

Reliable hybrid caching + database pipeline supporting real-time analytics and historical storage