

MongoDB Schema Design & Scaling Solutions

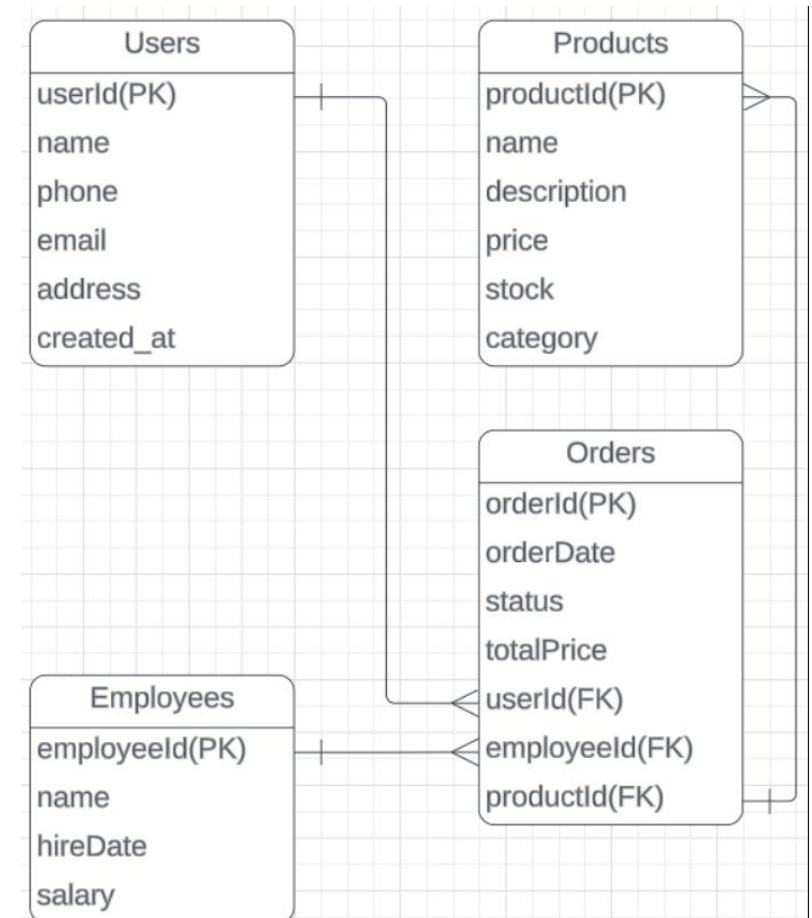
Proiect T3 – E-commerce

Objective Project

- Modelează date pentru domeniul ales (ex: e-commerce)
- Implementare CRUD: Create, Read, Update, Delete
- Aplica scaling și optimizare: indexuri, replicare
- Analizează compromisuri în schema design și scalare

Data Modeling

- **Entități principale:** Users, Products, Orders, Employees
- **Relații:**
- Order → User (referință)
- Order → Product (referință)
- **Strategii:**
- **Embedding:** date frecvent accesate împreună (ex: user address)
- **Referencing:** date mari / multiple relații (ex: orders, products)



Common Pitfalls in Schema Design & CRUD Operations

- Supra-embedding al documentelor → dimensiuni foarte mari ale documentelor
- Folosirea referințelor fără indexuri adecvate → interogări lente
- Nepreluarea validării datelor înainte de insert/update
- Stocarea datelor redundante în mod inconsistent
- Neglijarea gestionării scrierilor eșuate sau a interogărilor invalide

Indexing & Query Performance

Inainte de index: query lent pe totalPrice / orderDate

- Dupa index: query mai rapid
- Grafice / Timp executie: 0.05s → 0.04s

Trade-offs

Vertical vs Horizontal scaling:

- Vertical: simplu, rapid, limitat de hardware
- Horizontal: complex, scalabil, redundant

Schema embedding vs referencing:

- Embedding: read rapid, update greu
- Referencing: flexibil, read mai lent

MongoDB vs SQL:

- MongoDB excelent la volume mari și flexibilitate
- SQL mai bun la relații complexe

When MongoDB > SQL

➤ **MongoDB este mai potrivit atunci când:**

- schema se schimbă frecvent
- datele sunt ierarhice
- este necesar un throughput mare la scriere
- este nevoie de scalare orizontală

➤ **SQL este mai potrivit atunci când:**

- consistența puternică este esențială
- sunt necesare join-uri complexe
- schema este fixă

Concluzii

- MongoDB permite design flexibil și scalabilitate
- Alegerea schema + scaling depinde de dimensiunea dataset și trafic
- Indexuri + replicare + sharding cresc performanța, dar adaugă complexitate
- Practic: începe vertical + embedding unde e simplu, treci horizontal + referencing când crește traficul