



#ASLI ENGINEERING

Database per Service Pattern in Microservices



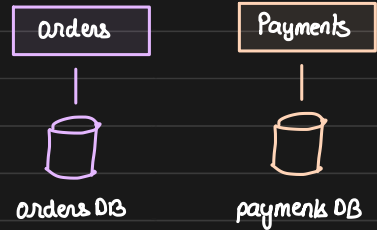
BY

ARPIT BHAYANI

Database per Service Pattern

Almost all microservices
need to persist data in some database

A database architecture we can use here
could be Database per Service



loose coupling is the highlight of microservices based architecture

Services should be

- independent to build, test, deploy, scale
- autonomous to take its own decision

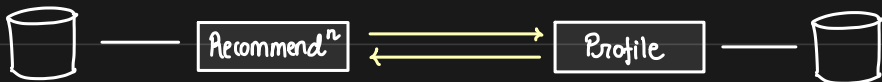
Database decisions play a key role here.

Social Network: Each service taking its own decision



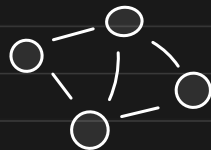
Advantages of going for Database per Service pattern

- you need **loosely coupled** components No direct access to DB



- you have a very **specific DB need** for your service

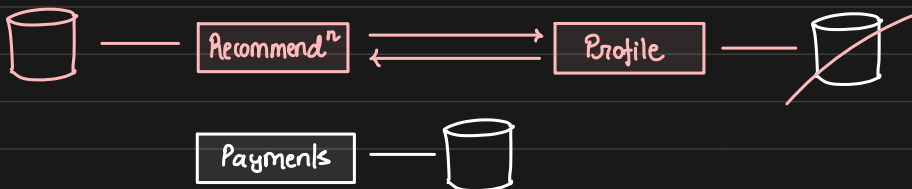
eg: graph db to model relations in social media



- you want **granular control and scaling** of your service

eg: horizontal, vertical, replica, partition, decentralized

- if a database goes down, it **only affects** the dependent services



- you have separate **compliance** need for a certain type of data

eg: encryption at rest for PII and financial data

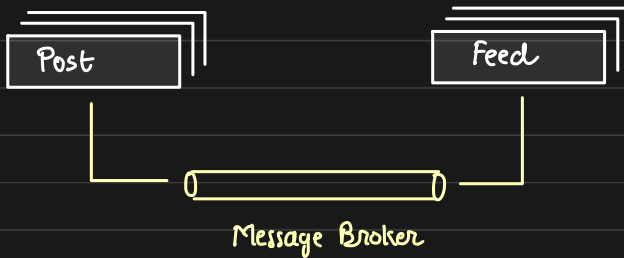
↳ doing it for 10GB table is simpler than 2TB of database

Disadvantages of adopting database-per-service

- cross-service transactions are **complex and expensive** ↗ time
→ throughput
eg: you will need to implement 2PC across services for TxN



- conveying updates across services is difficult



- multiple infra components to be monitored and managed