Datenbanken im Umfeld verteilter Systeme

psql

```
tickets=# CREATE TABLE tickets (
  id SERIAL PRIMARY KEY,
  title TEXT NOT NULL,
  status TEXT NOT NULL
);

CREATE TABLE comments (
  id SERIAL PRIMARY KEY,
  ticket_id INT NOT NULL REFERENCES tickets(id) ON DELETE CASCADE,
  body TEXT NOT NULL
);
CREATE TABLE
CREATE TABLE
```

Mongo

```
test> db.tickets.insertOne({
|   title: "Login schlägt fehl",
|   status: "open",
|   comments: [ { body: "Tritt nur mit SSO auf", author: "anna" } ]
| })
{
  acknowledged: true,
  insertedId: ObjectId('6abf54410639470122c3e003')
}
```

Vergleichsmatrix

Kriterium	Relational (PostgreSQL)	Dokument (MongoDB)
Datenmodell	Tabellen mit Zeilen und Spalten, normalisiert, feste Datentypen	JSON-ähnliche Dokumente (BSON) in Collections, verschachtelte Objekte und Arrays
Schema-Änderungen	Festes Schema, Änderungen per `ALTER TABLE` bzw. Migrationen	Schemaflexibel, Dokumente einer Collection können unterschiedlich aufgebaut sein; optionale Validierung per JSON Schema
Konsistenz & Transaktionen	Volle ACID-Transaktionen über beliebig viele Tabellen, starke Konsistenz	Atomar auf Dokumentebene; Multi-Dokument-Transaktionen möglich (seit 4.0), aber teurer
Beziehungen	Fremdschlüssel mit referenzieller Integrität, Verknüpfung per `JOIN`	Einbetten (Embedding) oder Referenzieren; `$lookup` statt JOIN, keine Fremdschlüssel-Prüfung
Skalierung	Primär vertikal; horizontal über Read-Replicas, Partitionierung oder Erweiterungen (z. B. Citus)	Horizontal eingebaut über Sharding und Replica Sets
Abfragesprache	SQL (standardisiert, deklarativ, sehr mächtig)	MongoDB Query API (JSON-basiert) und Aggregation Pipeline
Betrieb & Tooling	Sehr ausgereift: `psql`, pgAdmin, `pg_dump`, breite ORM-Unterstützung	`mongosh`, MongoDB Compass, `mongodump`, Atlas als Managed Service

---

Sources:

- https://bbw.iliraga.ch/321/presentation/#/70
- https://bbw.iliraga.ch/321/assignments/07-datenbanken-verteilte-systeme.html