**Vergelijking tussen Document Database (zoals MongoDb) en Graph Database (zoals Neo4J)**

**1. Structuur en data model**

* **Data model:**
  + Document Database: JSON/BSON-documenten met geneste eigenschappen en arrays.
  + Graph Database: Nodes en edges; relaties zijn first-class citizens.
* **Schema:**
  + Document Database: Schemaloos of flexibel schema → makkelijk toevoegen van nieuwe node types of properties.
  + Graph Database: Schema kan flexibel zijn, maar veel relationele traversals en types moeten worden gedefinieerd.
* **Relaties:**
  + Document Database: Relaties via arrays van references of embedded documents.
  + Graph Database: Relaties expliciet, direct verbonden met nodes.
* **Voorbeeld projectdata:**
  + Document Database: een Exo node met array van relaties naar Dof, Aim, Part.
  + Graph Database: Exo node met edges naar Dof, Aim, Part via HAS\_AIM, ASSISTS\_IN.

**2. Query mogelijkheden**

* **Filtering/search:**
  + Document Database: Zeer krachtig via MongoDB queries en aggregations.
  + Graph Database: Filtering kan, maar vaak via pattern matching.
* **Traversals:**
  + Document Database: Beperkt efficiënt bij multi-hop relaties (>2 stappen), 1-2 hop queries zijn snel.
  + Graph Database: Zeer efficiënt voor multi-hop queries (bijv. "vind alle Exos die via Dof een SKN beïnvloeden").
* **Aggregaties/dashboards:**
  + Document Database: Aggregation pipelines uitstekend voor statistieken en overzichten.
  + Graph Database: Aggregaties mogelijk maar minder intuïtief, gericht op netwerkstructuren.

**3. Webfrontend integratie**

* **JSON-native:**
  + Document Database: Perfect, direct bruikbaar in React/Vue/Angular apps.
  + Graph Database: JSON is mogelijk via drivers, maar vaak extra mapping nodig.
* **API:**
  + Document Database: REST / GraphQL makkelijk → JSON documenten.
  + Graph Database: REST / GraphQL mogelijk, maar complexer bij relaties en traversals.
* **Visualisatie (graph-view):**
  + Document Database: Kan via frontend libraries zoals D3.js of Cytoscape.js, relaties array-based gevisualiseerd.
  + Graph Database: Graph-view direct natuurlijk, nodes en edges kunnen 1-op-1 weergegeven worden.

**4. Schaalbaarheid en onderhoud**

* **Schaalbaarheid:**
  + Document Database: Horizontaal sharden, cloud-native (MongoDB Atlas, AWS DocumentDB).
  + Graph Database: Schalen is lastiger, clustering complexer.
* **Onderhoud:**
  + Document Database: Goed ondersteund, eenvoudig backups, flexibel voor updates.
  + Graph Database: Vereist kennis van graph operations; bulk updates lastiger.
* **Data import:**
  + Document Database: Bulk import eenvoudig uit Excel/CSV → JSON → MongoDB.
  + Graph Database: Bulk import mogelijk, maar mapping van relaties complexer.

**5. Flexibiliteit en uitbreidbaarheid**

* **Nieuwe node types:**
  + Document Database: Makkelijk toevoegen, geen schema migraties nodig.
  + Graph Database: Kan, maar vaak extra configuratie en edge-definities vereist.
* **Nieuwe properties:**
  + Document Database: Schemaloos → direct toevoegen aan documenten.
  + Graph Database: Schema soms aanpassen; minder dynamisch.
* **Nieuwe relaties:**
  + Document Database: Voeg nieuwe entry in relations array toe.
  + Graph Database: Edge definities toevoegen; meer setup.

**6. Kosten en projectimpact**

* **Setup en hosting:**
  + Document Database: Cloud-hosting eenvoudig (MongoDB Atlas, DocumentDB), goedkoper en minder complex.
  + Graph Database: Hosting kan duurder, clustering complexer.
* **Development tijd:**
  + Document Database: Sneller prototypen, eenvoudig iteratief uitbreiden.
  + Graph Database: Meer tijd nodig voor queries, schema-definities en traversals.
* **Frontend integratie:**
  + Document Database: JSON-compatibel → sneller ontwikkelen.
  + Graph Database: Extra mapping van graph queries naar JSON nodig.

**7. Conclusie: Waarom Document Based (MongoDB) de beste keuze lijkt**

1. Flexibiliteit: Nieuwe nodes en eigenschappen kunnen zonder schema-migraties toegevoegd worden.
2. Webcompatibiliteit: JSON-native, direct bruikbaar voor frontend, filtering, dashboards en graph-visualisaties via D3.js of Cytoscape.js.
3. Eenvoudige data-import: Bulk import van Excel → JSON → MongoDB makkelijker dan mapping naar Neo4j.
4. Traversals zijn beperkt: Projectqueries vereisen meestal 1-2 hop-relaties; MongoDB kan dit prima aan.
5. Schaalbaarheid en hosting: MongoDB Atlas/DocumentDB → makkelijk, cloud-ready, goedkoper dan Neo4j.
6. Aggregaties en dashboards: MongoDB Aggregation Pipeline krachtig voor statistieken zonder extra tools.
7. Onderhoud en ontwikkeling: Schemaloos → makkelijk uitbreiden, minder complex dan Neo4j bij updates.

**Afbeelding met tekst, schermopname, Lettertype

Automatisch gegenereerde beschrijving**