By Stefan Vasile and Jean-Paul Otten

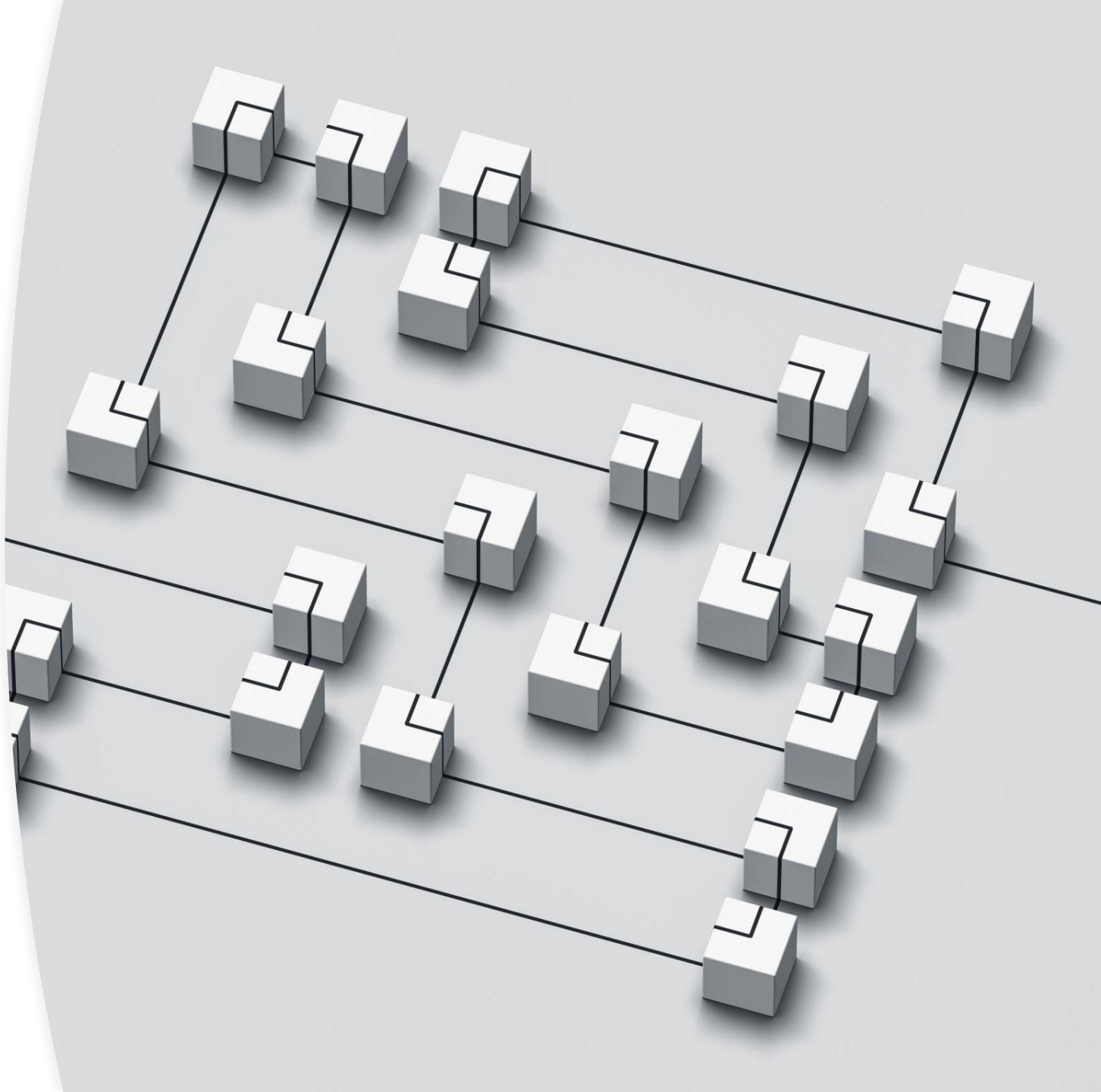
Workshop agenda

- Databases / DBMS
- Arango DB
- What problem does it solve?
- Relevancy?
- Pros and cons
- Competitors
- Questions?
- Workshop exercises

Databases or Database management system

- Which databases / database management systems do you already know?

- Which ones have you worked with?



Arango DB

Initial release in 2011

Written in C++

Stores Json based data

Multimodal DBMS

What problem does it solve?

Unify different data representations

Model data Flexibly

One query language for all

Why is it still relevant?

- Industry trends
- Multiple data model types
- One database

Pros and cons

- Users appreciate the **ease of use** of ArangoDB, finding its intuitive AQL language and versatility very user-friendly. (17 mentions)
- Users value the **flexible schema and powerful querying** features of ArangoDB, enhancing data management and performance. (12 mentions)
- Users praise ArangoDB for its **intuitive AQL language**, facilitating quick learning and efficient complex query execution. (8 mentions)
- Users appreciate the **intuitive AQL for queries**, enjoying its simplicity and flexibility for developers and complex operations. (7 mentions)
- Users appreciate the **customization capabilities** in ArangoDB, allowing for flexible data models and intuitive querying. (6 mentions)
- Users value the **flexibility** of ArangoDB, benefiting from its versatile multi-model approach and intuitive AQL language. (6 mentions)
- Users benefit from **responsive and helpful customer support** that enhances their experience with ArangoDB. (5 mentions)
- Users appreciate the **intuitive web interface** of ArangoDB, making database management straightforward even for newcomers. (5 mentions)

- Users note a **lack of information** on ArangoDB's documentation and limited support for Python and BI connectors. (7 mentions)
- Users find that **improvements are needed** in intuitiveness, editor quality, and performance of ArangoDB's features. (6 mentions)
- Users find the **poor usability** of ArangoDB challenging, citing issues with the editor and lack of intuitive functions. (6 mentions)
- Users face a **steep learning curve** with ArangoDB's AQL, making complex queries challenging to execute effectively. (5 mentions)
- Users experience **slow performance** with ArangoDB, particularly during complex aggregations and graph traversals on large collections. (4 mentions)
- Users find the **learning curve steep**, especially when transitioning from SQL to ArangoDB's AQL for complex queries. (3 mentions)
- Users find the **poor documentation** of ArangoDB challenging, impacting their ability to utilize advanced features effectively. (3 mentions)
- Users find ArangoDB **expensive** for simple CRUD apps, suggesting more cost-effective alternatives for such use cases. (1 mentions)

Specific comments on ArangoDB

Talk about being delusional. My company bought the hype and switched from SQL to Arango. Just investigate write-write conflicts. The DB is garbage. Don't worry about that error, or the others that require indefinitely retrys, it locks up under heavy load. We actually hit a crash in the engine, so now we are at the mercy of ArangoDB to fix our production DB.

I have been working with ArangoDB for a couple of years now and have been very happy with that [illegible] since.

For me memory has never been an issue. I'm working with large amounts of documents, large [illegible]s and complex queries that address large amounts of documents in multiple collections. Never ran into any memory trouble or seen any issue related to memory. Maybe the memory comes into play when you will work with vast amounts (tens/hundreds of millions), but that i don't know.

I decided to go with Arango because of the strong feature set.

A very important reason was the very powerful query language (AQL) which you can extend yourself trough user-defined functions.

Basically in my setup i want to use JSON storage over http for serving one page apps and did not want to lose any functionality i had when working with an RDBMS and write my own API. Really the only acceptable choice for me back then was ArangoDB. Might be that others added a lot of similar features by now but my first show stopper with others on the list (Mongo, Couch, Rethink) was the query language.

There is one thing i'm really missing tough and that is websockets. But it's on the list somewhere, hopefully that will be added at some point.

So i'd say, it's of course very much depending on your use and the memory concerns i cannot really tell you about other than, never seen any troubles, but from (my) one page app perspective it's a great choice.

Competitors

Document based

Key value store

Graph database

Category	ArangoDB	Neo4j
Data model	Multi-model (documents, key-value, graphs)	Native graph database
Query languages	AQL (unified across models)	Cypher (pattern-matching optimized)
Performance	Great for hybrid workloads; Cross-shard graph traversals can be slower	Best-in-class for deep and dense graph traversals
Scalability	Native horizontal scaling; SmartGraphs for locality	Causal Clustering; Fabric for federation
Analytics	Basic analytics capabilities plus Pregel-based graph algorithms	Full Graph Data Science (GDS) suite for graph ML, algorithms, and analytics
Operations	Simpler operational model when multiple data models are required	Stronger governance and enterprise tooling for large graph-centric deployments

Neo4j vs ArangoDB


Questions?

Workshop exercises

Github link:

https://github.com/FontysVenlo/esd-workshop-arangodb_esd

Questionaire

https://kahoot.it/