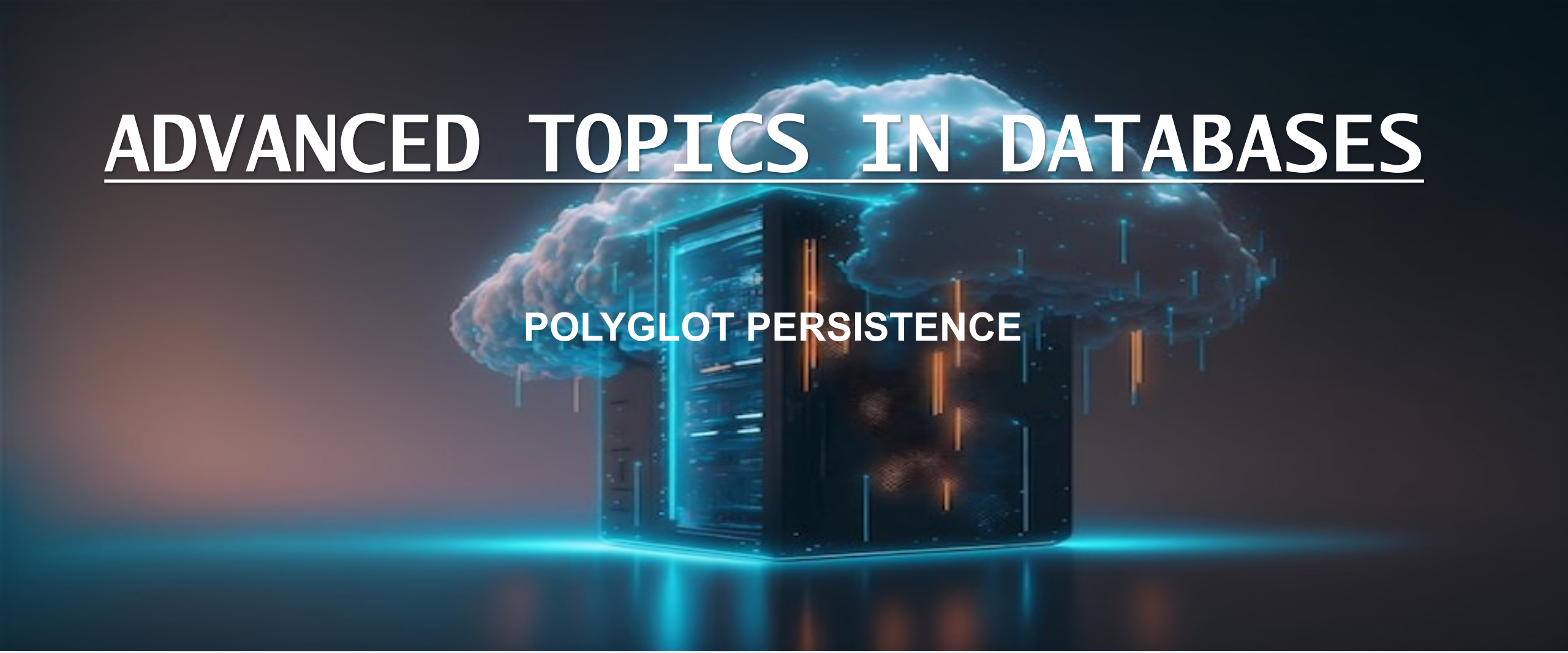


ADVANCED TOPICS IN DATABASES

POLYGLOT PERSISTENCE

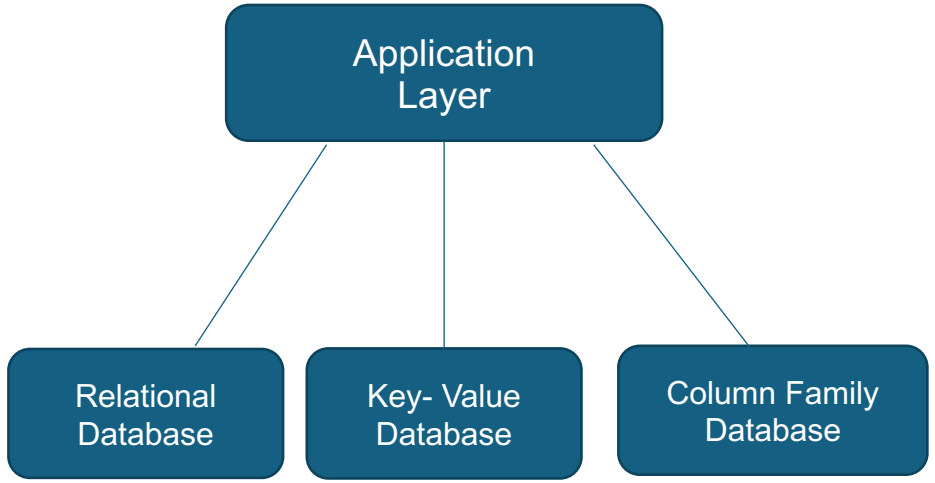
Master in Informatics Engineering
Data Engineering

Informatics Engineering Department

ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

Polyglot Persistence

- A set of applications that use s**everal core database technologies;**
  - **Can** will also occur within **a single application** as different parts of an application's data store have access characteristics;
- With polyglot persistence**, all requirements can be satisfied by using as many databases as neede**d.



Architecture

- if the application **is not small,** it would be **divided into modules or components.**
  - Each module is responsible for part of the application and has its own logic and functions.
    - If the **data managed by a module is not shared by any other module**, managing polyglot persistence would be simple because each module can have a different exclusive database system.
    - If **data are shared by more than one module**, can be **conflicts of requirements** (that must be satisfied using more than one database system).

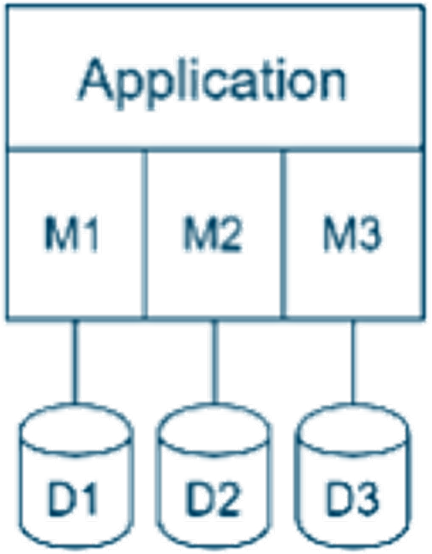
Architecture

- **An application** can **have a combination of these** types of relationships:

one- to- one

1- A module has a connection with one exclusive DB;

2- A failure of one database will affect only part of the application and will not be propagated



Architecture

- **An application** can **have a combination of these** types of relationships:

one- to- many

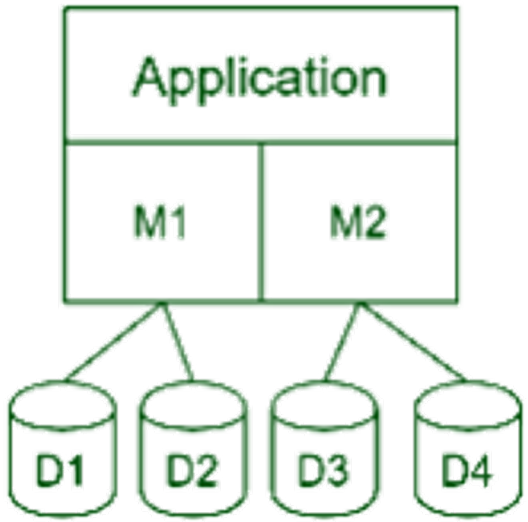
1- A module has a connections more than exclusive DB;

2- May make the programming task more confusing. Database consistency needs to be managed by the module;

3-running a query across different databases is not straightforward

4- **redundancy might be a problem** when **different parts of the same data object are stored at different databases**

5- **a failure of one database**, this failure might be **logically cascaded to other databases**



Architecture

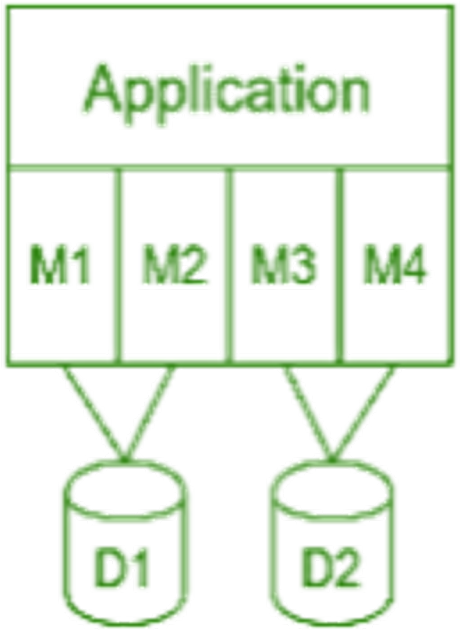
- An application can have a combination of these types of relationships:

many- to- one

1- More than one module have connections with one mutual DB;

2- If all modules of the application are using the same database, then the polyglot persistence concepts are not applied.

3- database configurations and security depend on more than one module, which may violate the Least Common Mechanism security design principle

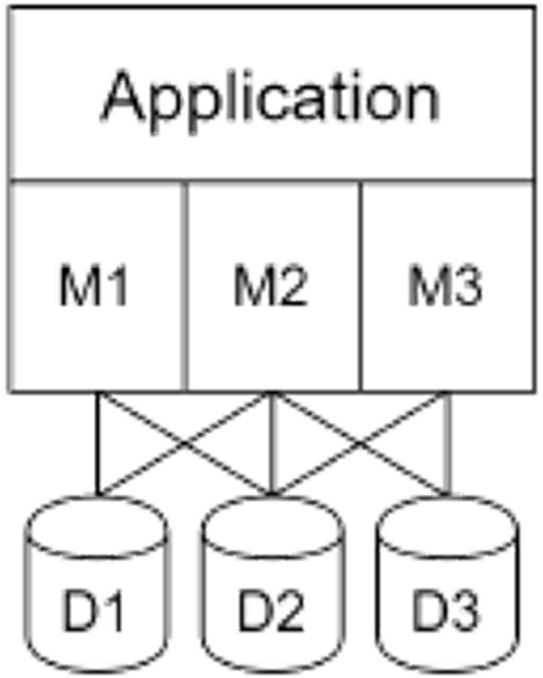


Architecture

many- to- many

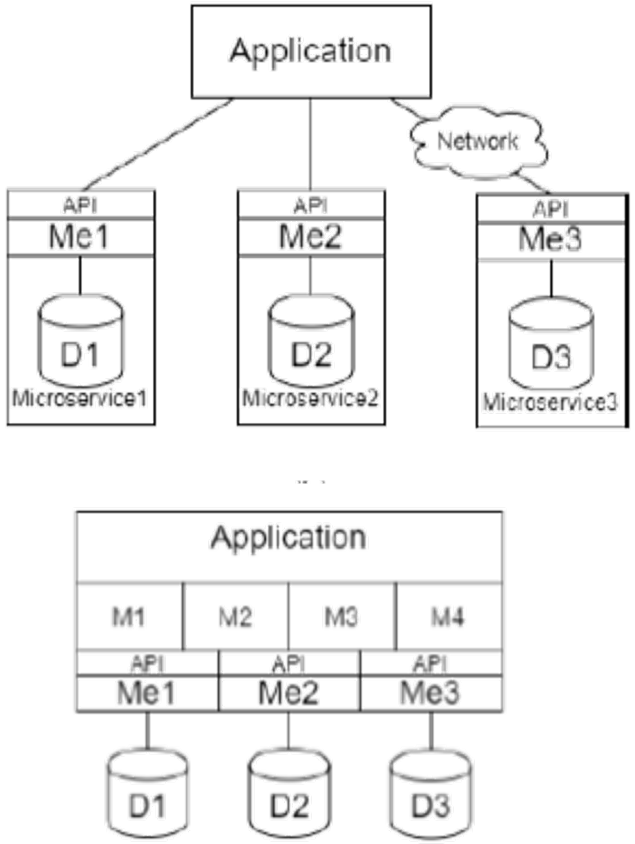
1- More than one module have connections with more than mutual DB;

2- there will be a need to maintain cross-modules consistency for the data objects that are dependent on each other and stored in different databases



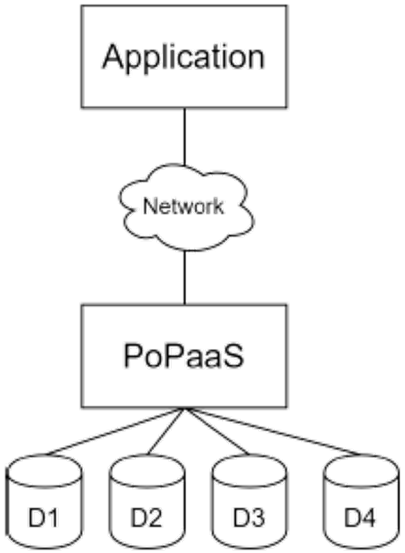
Architecture

- **Persistence Service-oriented** - If the database is being used by more than one module or application application, then the database can be decoupled from the application to reduce the complexity
  - **Mediator** - will be able to access and control the database. This mediator **is an independent module** or **a small application that offers an API for external users.**
    - completely be independent from the application and can be deployed in a different machine - **the mediator is part of what is called a microservice**
    - the mediator is part of the application, and it offers an API as public function calls to other modules, or even other applications.



Architecture

- **Polyglot-Persistence-as-a-Service**
  - managing polyglot persistence is conveyed to a completely different location - Cloud;
    - The application here only provides the data requirements, and then these requirements should be satisfied by the Polyglot-Persistence-as-a-Service (PoPaaS) provider.
  - **Problems:**
    - is how the client can formulate the requirements in a standard format;
    - is the selection of the appropriate database system for the given requirements, which should be automated based on quantifiable metrics

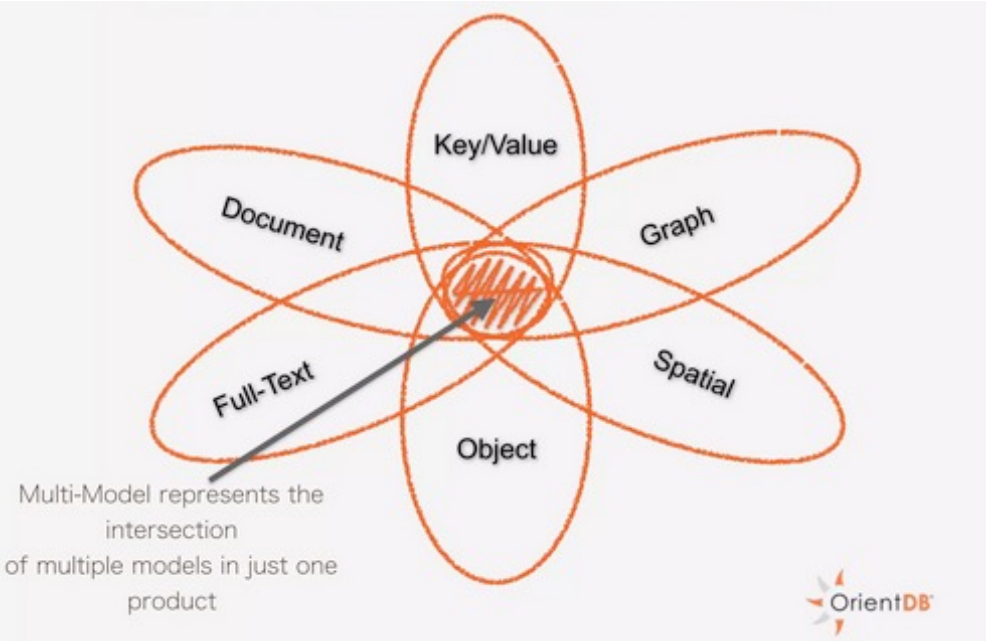


Architecture

- **Multi-models Databases**
  - is to use a database that supports multiple data models, which is called a multi-model database.
    - the application will manage a single database that fulfills its requirements.
  - has the features of several data models.

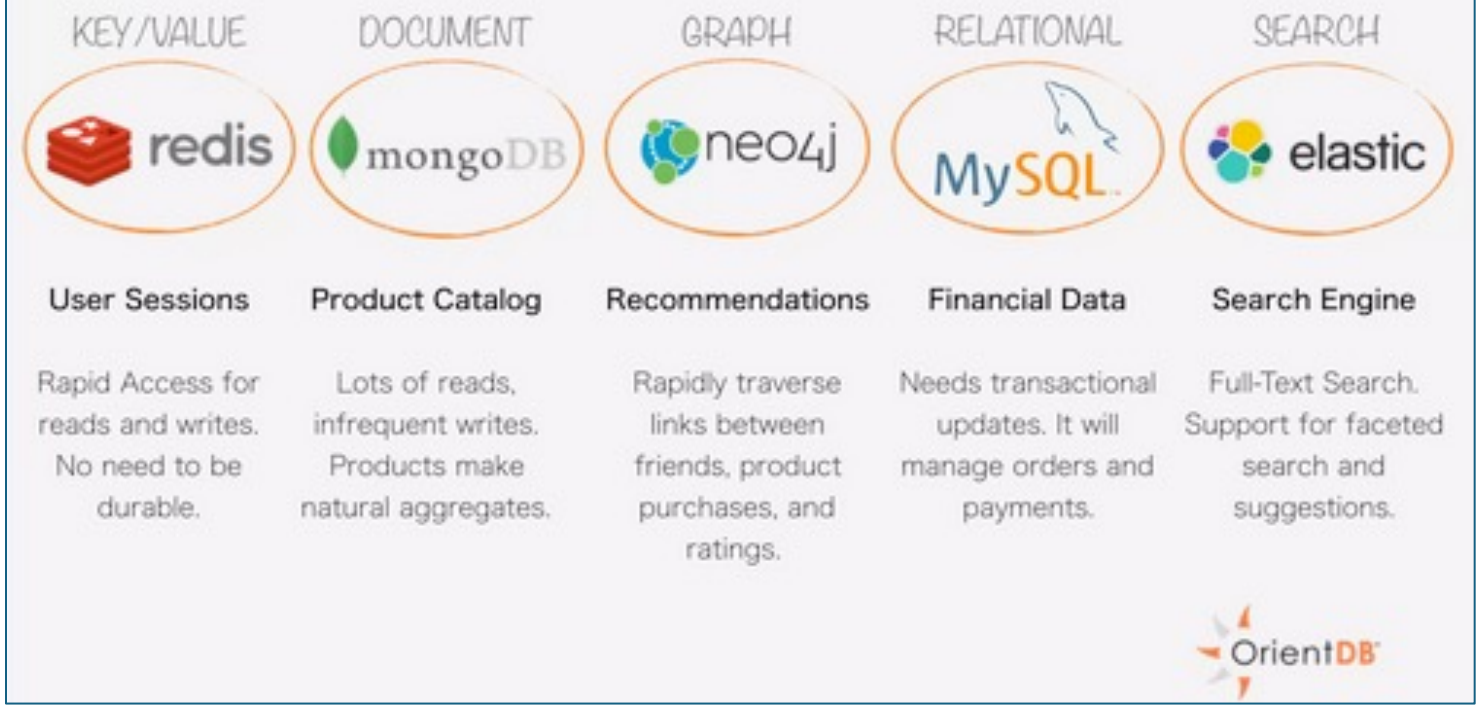
Examples:

OrientDB, ArangoDB, Microsoft Azure Cosmos DB and Amazon DynamoDB



Polyglot Persistence in Action

Example: Hotel Booking Application



Polyglot

KEY/VALUE

redis

User S

Searc

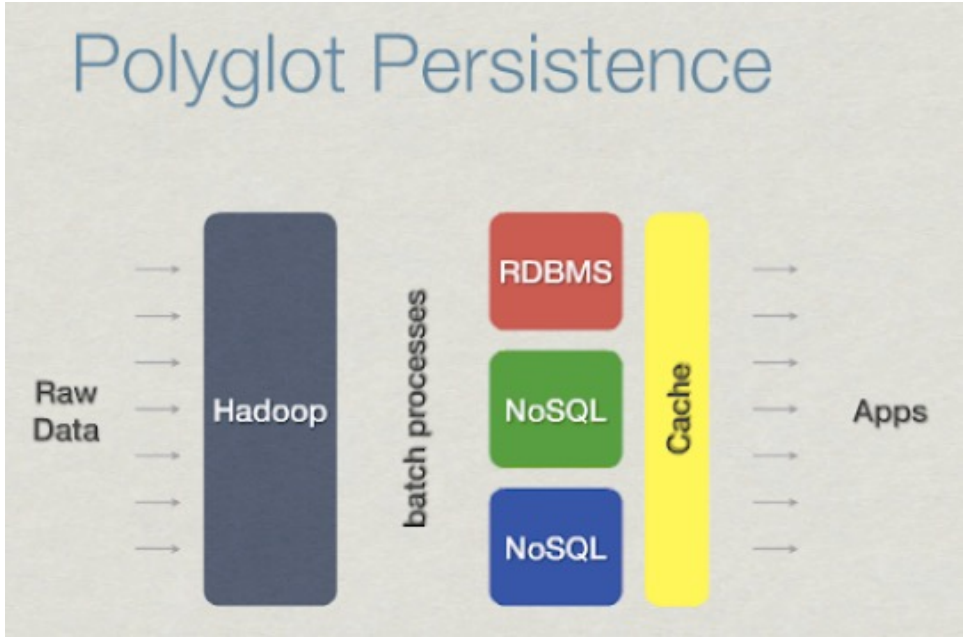
Polyglot	Multi-Model
• 5 products to learn • 5 servers to configure and deploy • 5 vendors in case of support	• Only 1 product to learn • Only 1 server to configure and deploy • Only 1 vendor in case of support

OrientDB

OrientDB

Key Points

- **Polyglot persistence** is about **using different data storage technologies** to handle varying data storage needs.
- Polyglot persistence can apply across an enterprise or within a single application.
- **Encapsulating data access into services** reduces the impact of data storage choices on other parts of a system.
- **Adding more data storage technologies increases complexity in programming and operations**, so the advantages of a good data storage fit need to be weighed against this complexity.



Implementation Methodology

1. Database Requirements must be consistence with application requirements.
   - Include functional and non-functional and data requirements
   - Design a conceptual data model.
     - Can be Domain Requirements (UML)

2. Database Selection - Conflicts between database requirements should be identified and resolve by using as many database systems as need
   - A clear mapping between each part of data and the selected systems should be preserved

The step 1 and Step 2 can be accomplished using a systematic approach

Systematic Approach

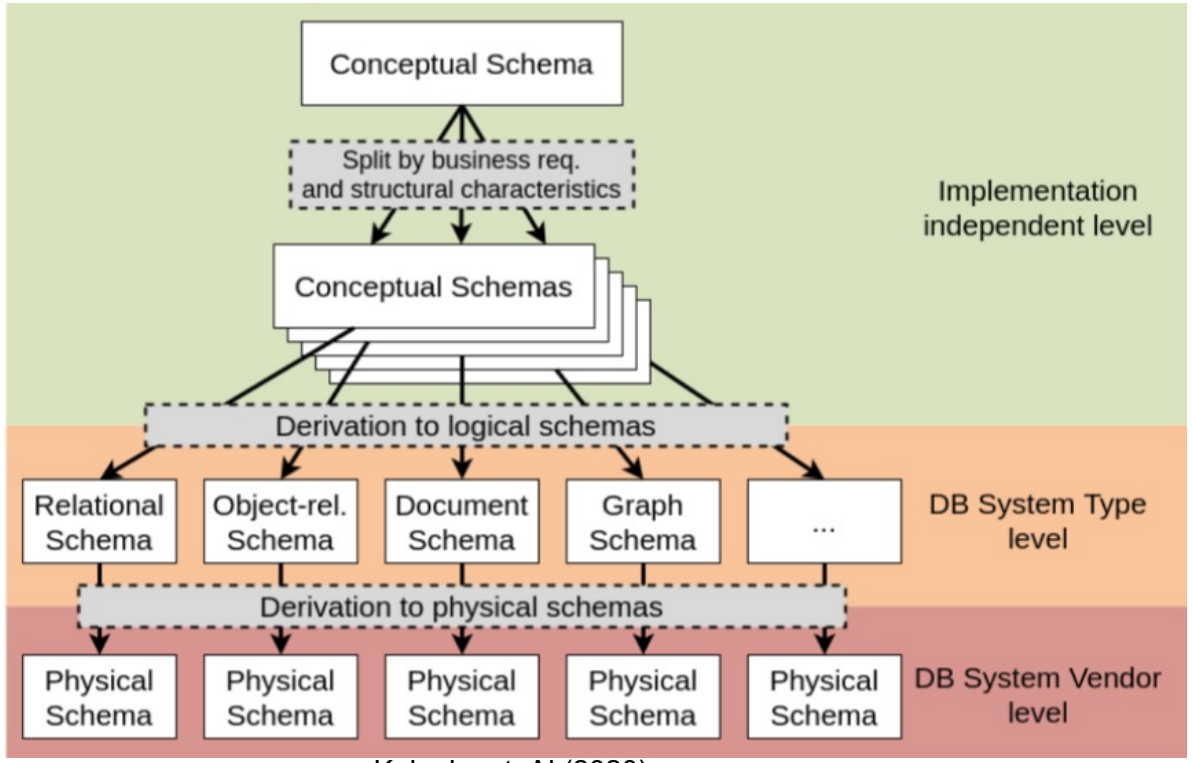
The method consists of the following steps:

by Roy-Hubara et al. 2022

1. Gather and specify the data-related requirements and express them via a conceptual data model.
2. Gather and specify the functional requirements that are related to database operations, i.e., data retrievals and update operations; hereafter, we call them queries.
3. Gather and specify the non-functional requirements that are related to the data-related requirements and the queries.
4. Based on the above, consider dividing the conceptual data model into fragments, each of which may have different characterizations (e.g., different performance and consistency requirements). The result of this step may be the selection of more than one database model.
5. Select the most suitable database model for each fragment. This will be based on a general-purpose pre-defined profile of each database model. A pre-defined profile consists of a set of non-functional properties associated with each database model

Implementation Methodology

1. Database Specific Data Models - Since different database use different storage models, each of them require a specific model to determine database schema.



Kolonko et. Al (2020)

Implementation Methodology

2. Polyglot Persistence Architecture – The architecture determines how the application will be talking to the different selected database.
   - The architecture design should considerer implementation cost, time and, resources.

3. Issues Identification - Associated with the selected architecture should be identified.
   - A plan should be made which include technical details on how each issue will be addressed (how they will be resolved)

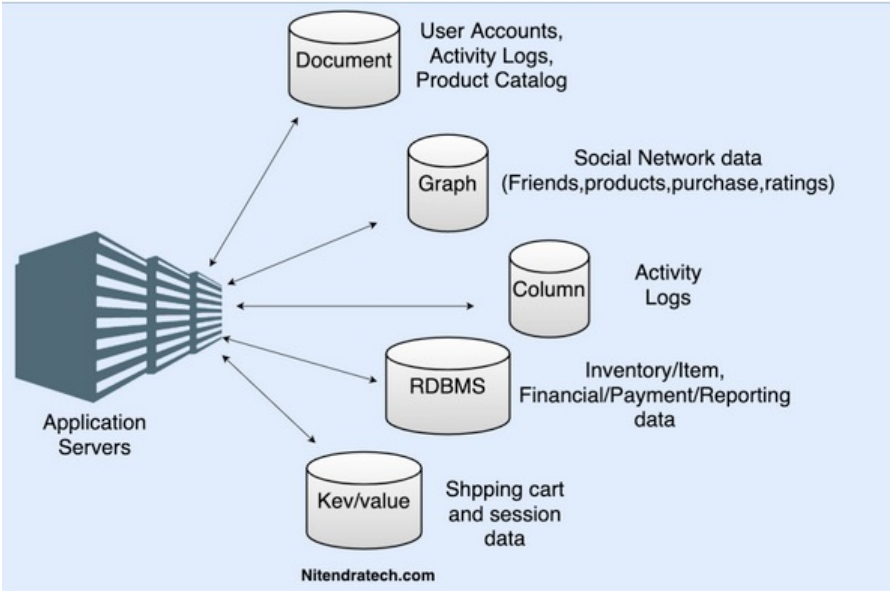
4. Application Development - should start based on the results of the previous steps;

When to Use NoSQL

- It depends on factors like….
  - data is mainly collected or displayed in terms of aggregates
  - data include complex, nested, or hierarchical structures
  - data has a lot of relationships (graph databases)
  - data is no-uniform
  - The Database logic can be encapsulated into an isolated section of project
  - data access performance (faster)
    - data need to be clustered (fragmented and/or replicated)
    - aggregate data would need to be joined from multiple tables in RDBMS
    - complex relational data needs to be queried (graph database)

When is Polyglot Persistence Pertinent?

- Application essentially composing and serving web pages
- only lookup page elements by ID
- different needs or availability, concurrency and no need to share all their data
- Scaling to lots of traffic gets harder and harder to do with vertical scaling
- Many NoSQL databases are designed to operate over clusters.
  - They can tackle larger volumes of traffic and data than is realistic with a single server



HINTS

If we have structured data with some differences - use a document store

If we have relations between and want to efficiently query them use a graph database

If we have structured data where all objects have equal attributes use relational database

Readings

Slides based on:

1- Omar Lajam and Salahadin Mohammed, “Revisiting Polyglot Persistence: From Principles to Practice” International Journal of Advanced Computer Science and Applications(IJACSA), 13(5), 2022. http://dx.doi.org/10.14569/IJACSA.2022.0130599

2- M. Kolonko and S. Müllenbach, "Polyglot Persistence in Conceptual Modeling for Information Analysis," 2020 10th International Conference on Advanced Computer Information Technologies (ACIT), Deggendorf, Germany, 2020, pp. 590-594, doi: 10.1109/ACIT49673.2020.9208928.

3- Pramod J. Sadalage, Martin Fowler, “NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence”, Addison-Wesley, 2013 (Livro).