STYLE OF DATABASE

Data storage technologies according to operational needs: transaction, integrity, concurrent access, history, data natures; volume, governance, etc. The choice of these technologies is important for deciding the architecture of the semantic platform and more specifically the MDM, ODS, and EKG repositories.

1. CONDITIONS OF SUCCESS

AI systems need to be integrated with semantic data management; otherwise, the training processes weaken and profitability does not materialize. It is thanks to metadata and ontologies that AI better understands the meaning of information. Generally, the quality level of the data provided to AI conditions the level of intelligence obtained at the end of their training and execution.

In this context, the choice of database technologies to successfully implement AI is fundamental. It takes into account these four essential needs for obtaining high-performing AI systems:

1. **Data labeling**: AI learning processes rely on metadata that serves as labels describing their usage context. For example, the metadata of a bank credit file provides the history of its subscription, the calculation of its score, and the relationships to business concepts such as the client and the financed asset. The boundary between metadata and operational data is not always stable. In practice, metadata exists through ontologies, that is, unified data models to be implemented in the semantic platform as described by TRAIDA, with MDM, ODS, and EKG repositories. Therefore, their management must be intelligently integrated with production databases and shared ontologies at the enterprise level.
2. **Description of multimedia data**: Documents (file, image, video, text…) are enriched with metadata that helps AI systems interpret them. They also document the relationships that exist with the business concepts operated by the company. For example, a client email is classified according to the nature of the request and attached to the client file.
3. **Data grouping for AI system training**: The training process of an AI requires injecting datasets of different formats and origins. For example, an AI assistant for customer relationship support is trained with product descriptions, a user guide from the online order website, an ebook published by the company, the FAQ, etc. This set of files must be kept in an archive to retain the memory of the training carried out. It will be necessary to audit the functioning of the AI and for unlearning processes when certain outdated or erroneously loaded information needs to be removed from the AI.
4. **Data injection in AI conversations** (with the RAG - Retrieval Augmented Generation technique): This involves enriching the content of AI queries with access to databases. For example, submitting a ChatGPT prompt about a client file automatically generates a read in a database to retrieve the most up-to-date client information. Thus, the AI accesses information beyond the data already injected at the time of its training. This injection principle is also used to verify and complete the response formulated by the AI; it is then an interesting way to detect hallucinations and trigger alert and correction processes.

For these use cases to be successful, the company's data must be organized in high-quality repositories. In other words, using AI on a large scale while having poor data management is doomed to fail. To avoid this impasse, data preparation and governance rely on a semantic platform as described in TRAIDA. It implements three strategic repositories: MDM, ODS, and EKG. These are positioned downstream from operational applications and upstream from decision-making systems (data warehouse, data lake).

The data in these repositories must be of the highest possible quality. To achieve this, they share the same metadata and ontologies across the enterprise. This unification facilitates their interpretation for AI training. The semantic platform ensures the management of these ontologies.

In this context, choosing database technologies to meet the needs of AI and ensure integration into your information system that considers your specific constraints is essential. Therefore, depending on your requirements in transactional management, the nature of the data handled (structured data, multimedia), exploitation needs (unitary, collective, multidimensional, search...), volume and intensity of concurrent access, and maintenance agility, several technical approaches are available and complementary. These are described in the following sections of this card.

2. IMPORTANCE OF THIS CARD FOR YOUR TRANSFORMATIVE AI

Data management technologies form the foundation of the semantic platform for AI. It is necessary to reconcile both efficiency and agility criteria. In other words, it is not enough to choose a high-performance technology if, at the same time, it leads to system rigidity. Conversely, it is useless to opt for a flexible solution if it does not allow for scaling across the entire scope of the enterprise.

Depending on your context, you will likely need to choose multiple technologies for MDM, ODS, EKG repositories, as well as for data warehouses and data lakes. These strategic repositories can also be accompanied by more tactical ones included in data hub and data fabric solutions (see the TRAIDA "Data Integration" card). Depending on your needs for structured and unstructured data (multimedia) volumes, as well as your transactional management requirements and simultaneous multi-user access levels, the technological choices will differ.

The most fundamental way to categorize database technologies is based on the data schema management mode, that is, how the data model is described. We will now review these categories.

STRICT SCHEMA

With strict schema technology, the data description is formal, complete, and deterministic. For example, this involves describing tables, fields, and relationships for a relational database. A data description language is used, which also allows for expressing integrity and quality control rules. Since the data structure is explicitly described, this technology enables transactional management (ACID). Additionally, it allows for response time optimization through the creation of specific indexes.

The constraints of this approach are as follows:

- Little to no capacity to store and manage multimedia data whose structure is difficult to predict. Files are then referenced as simple links to third-party storage technology.
- Lack of agility for the process of modifying data structures. This maintenance requires technical intervention at the level of the data schema description, followed by redeployment of the database. This is a delicate IT expert task that does not allow for involving business teams.
- Since there is no predetermined data schema, database governance functions are limited to technical processes, such as backup and API exposure.

Examples of solutions: Oracle Database, Microsoft SQL Server, PostgreSQL.

META SCHEMA

This approach relies on a strict data schema that can describe any other data schema. In a minimalist approach, this meta-schema could be reduced to a single object (or table) with a reflexive association. For example, a Client object is then represented as a main record, and each of its fields (or columns) is represented by another secondary record (child of the main record). In reality, the meta-schema is more complex and richer than a simple object with a reflexive relationship. Nevertheless, its structure is ignored by technical and business teams who use the database through a data modeling tool. The internal system of the database then handles the mapping between business models and the meta-schema.

The agility of this approach is much better than that of the strict schema. Indeed, the creation or modification of a new data model is done declaratively in the modeling tool without the need to technically intervene at the level of the internal schema of the database. Moreover, since the data model is dynamically introspected by the database, it becomes possible to provide advanced governance functions whose behavior adapts to the semantics of the data.

However, the following weaknesses must be taken into account:

- Multimedia data that does not have predefined structures cannot be managed intrinsically. Only links to third-party storage systems are feasible.

- Data quality control rules are no longer integrated into the internal data schema of the database, as is the case with the strict schema approach. Therefore, these rules must be entrusted to an application layer whose reliability and performance depend on the quality of the developments.

- In the absence of a dedicated data schema for each data model, it is no longer possible to optimize it specifically. Only optimizations applied at the meta-schema level are ensured and thus escape the control of the database administrator. Depending on the data volumes managed and the specifics of certain access queries, performance issues may arise and hit an optimization barrier.

- Given the lack of optimization, the behavior of the database for transactional management (ACID) on large volumes and in the context of intense multi-user access must be carefully verified. The results are generally not as good as those obtained with the strict schema approach.

Examples of solutions: Apache Hive, Microsoft Azure Data Catalog, Informatica Metadata Manager.

DOCUMENT SCHEMA

This storage technology focuses on multimedia data that does not have predetermined structures. The term "document" aptly describes the organization mode of stored information, in the form of simple documents (JSON, BSON, or XML, and binary files). Therefore, it is not necessary to create a data schema since the generic concept of a document prevails. In this sense, it is also a meta-schema approach, but this time dedicated to multimedia data.

Depending on the database used, it is possible to attach descriptive metadata to the documents in the form of a list of keys and values. The advantage of this technology is the rapid and massive storage of multimedia data without the need for a modeling step. It is common in big data projects.

The main weaknesses to note are:

- Little or no querying capability that spans multiple documents. This is a corollary of the absence of relationship management. To implement querying between a parent document and child documents, it is necessary to denormalize the storage of related documents by duplicating them in each branch of the lineage.

- No introspection of the content of the documents. The database is limited to storing them in an atomic manner. Only descriptive metadata can accompany the document to describe it.

- Little or no propensity to store structured data, and thus no transactional management or management of relationships between documents.

Examples of solutions: MongoDB, CouchDB, Amazon DocumentDB.

GRAPH SCHEMA & SCHEMA FREE

With knowledge graph-oriented database technology, the storage mode is similar to that of the meta-schema. This relies on a definition of triplets consisting of a start node, a relationship, and an end node. This is known as the graph schema.

For example, modeling a customer who orders a product is expressed in the form of two nodes for the business concepts Customer and Product, connected by a relationship that expresses the act of purchase.

Unlike the meta-schema approach described earlier, it is not mandatory to model the data structure. More precisely, it is possible to activate a "schema-free" mode that generates triplets on the fly based on a data source whose structure is discovered at the time of loading. In this case, the quality of the obtained triplets depends on the quality of the injected data. This mode of operation amounts to creating an initial version of a data model from an information source. For example, with the help of a generative AI like ChatGPT, it is possible to extract business concepts and their relationships by introspecting the content of documentation, then automatically generate the knowledge graph without having to go through a preliminary modeling step.

Thus, the source document, for example, a PDF file of financial regulations containing several dozen pages, is visualized in the form of a knowledge graph. By injecting a catalog of metadata into the AI to help identify the business concepts of the company, this graph will correspond to a version close to the final result. The TRAIDA card that deals with "core system data" also addresses this use case, applied to the analysis of information system data. You can refer to it for additional information. The advantage of the graph schema and schema-free approach lies in its flexibility when transitioning from one to the other. Knowledge graph-oriented database technology thus offers the best of both worlds:

1. Explicit modeling with the graph schema that remains generic with meta-storage in the form of triplets.

2. Schema-free mode that does not require modeling and is similar to document storage. However, it is not about storing multimedia documents but structured triplets.

The disadvantages of the graph schema and schema-free approach are as follows:

- This technology is more suitable for managing structured data, although some solutions also claim to handle multimedia data. In practice, beyond marketing intentions, this essentially involves adding a third-party storage solution, such as a file system or cloud storage like Amazon S3, Google Cloud Storage, Azure Blob Storage, or directly a document-oriented database (see above).

- As with the meta-schema approach, quality control and referential integrity management rules are relegated to an application layer, which can suffer from malfunctions and performance issues.
- Similarly, transaction management (ACID) is an overlay to the storage technology, which can also experience malfunctions at the limits of volumes and simultaneous user access.
- Specific optimization of the data model is not possible, and it is necessary to rely on the quality of the technical solution used.

Examples of solutions: Neo4j, Amazon Neptune, ArangoDB.

VECTOR DATABASE

This is an additional storage area to the databases we have previously discussed. It is necessary for providing information to AI systems.

Vectorized database technology is not dependent on a specific type of data schema. In other words, there is no need for a preliminary data modeling step. It is sufficient to inject a data source into the vectorized database. From then on, it becomes available to load information for AI system training, as well as for on-the-fly access such as Retrieval Augmented Generation (RAG).

Generative AIs like ChatGPT ensure the storage of data in a vectorized form. They handle the process of loading data sources into their own vectorization technologies, eliminating the need for manual intervention.

Examples of solutions: Pinecone, Milvus, Weaviate.

FULL-TEXT SEARCH DATABASE

Full-text indexing technology complements database queries (such as SQL) with full-text and phonetic searches. The data from the database is then injected into an indexing technology dedicated to these types of searches. This allows for set-based searches that navigate through all the data and their relationships in a performant manner. However, it is necessary to establish a technical integration with the source databases to ensure that the indexes are updated at a frequency that meets the company's needs.

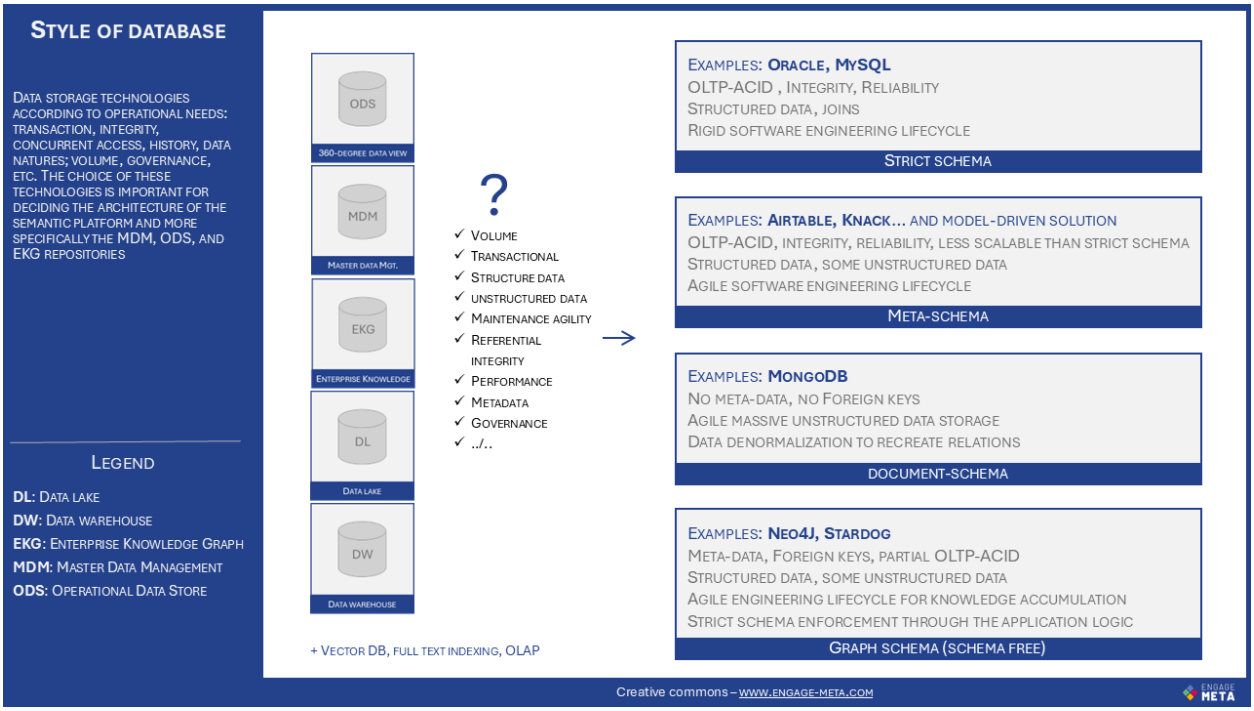
Examples of solutions: Elasticsearch, Apache Solr, Algolia.

ANALYTICAL DATABASE SCHEMA

In the realm of decision-support computing, there is a need for multidimensional data analysis. This involves navigating all the relationships between data from a point of interest without having to formalize a query that could be complex. Visual navigation is required, with progressive levels of data aggregation according to various dimensions. The implementation of this type of solution relies on OLAP (Online Analytical Processing) technology, which organizes the data in the form of cubes (stars) or snowflakes (fractals).

Examples of solutions: • Snowflake, Google BigQuery, Amazon Redshift.

3. BLUEPRINT



4. YOUR SITUATION & OBJECTIVES