TRAIDA Quiz Answers

Transformative AI & Data Solutions

December 15, 2025

Content

This deck presents the answers to the questions listed in the **TRAIDA Quiz deck**

01

During the business data modeling process for AI at which levels does metadata exist?

- Metadata is first captured at the Business Glossary level, which defines and standardizes business terms
- The Business Data Model then constitutes a structured repository of metadata, describing how data entities relate to one another. It is important to recall that semantic clarity largely resides in the relationships between data, not only in the data elements themselves
- Finally, data use-case descriptions (including narratives and synthetic data) can be considered a meta-knowledge layer that helps AI systems understand data and their usage context (Semantic Layer)
- This leads to a deeper understanding of the enterprise's information assets and represents an essential formalization level for reducing AI hallucinations

02

How can tacit knowledge be transformed into explicit knowledge?

- Knowledge that is not formally documented is referred to as tacit knowledge. It resides in individuals' experience and is primarily transmitted orally within the organization
- To properly train AI on real work practices, it is essential to convert tacit knowledge into written form. This documentation effort requires knowledge formalization skills, which implies training teams in structured writing. Today, this capability is still insufficiently mastered, and on average even less so among younger generations of workers
- To support the transformation of tacit knowledge into explicit knowledge, it is recommended to establish an AI-supported knowledge writing process. Such a process enables both individual and collective knowledge accumulation, notably through the facilitation of collaborative working groups
- On average, around 70% of enterprise knowledge is not formalized in information systems or written documentation. Yet, this knowledge represents a critical raw material for effective AI training and for achieving a measurable ROI from AI-driven automation
- Within TRAIDA, the WASI concept (Write – Analyze – Share – Innovate) helps build individual capabilities for knowledge formalization and long-term accumulation

03

How does a Service-Oriented Architecture (SOA) enable better AI deployment?

- The Service-Oriented Architecture (SOA) approach is applied at two levels. First, it is used to define the pre- and post-conditions required to control the execution of Large Language Models (LLMs). These conditions implement guardrails to secure input and output data flows, as well as to detect and handle hallucinations. When strict and verifiable control of pre- and post-condition rules is required, these mechanisms must be implemented using formal programming languages that do not rely on LLMs
- Second, SOA is used to build the Semantic Layer, exposing business APIs to AI agents rather than direct physical data access flows. The business services provided by the Semantic Layer handle the orchestration of logical and physical access to databases. AI agents consuming these services therefore interact with the enterprise business vocabulary, which significantly reduces hallucination risks through alignment with the Business Glossary

04

How does an Event-Driven Architecture (EDA) improve AI deployment?

- Inter-system exchanges are based on the principle of event publication to ensure loose coupling between systems. In other words, when system A needs to call system B, it publishes an event in the exchange layer, which will then be retrieved by system B
- Similarly, at the intra-application level, development relies primarily on the “Record Life Cycle Design” transaction management pattern rather than the classic ACID pattern spanning multiple tables. Only transactions that require a high level of security are based on the ACID pattern
- Finally, the use of workflows (BPM) should be limited to a small number of sensitive business processes to avoid organizational rigidity. It is preferable to implement stateless workflows based on status update management, leaving human actors to act according to needs and procedures in place within the company. An automatic monitoring and auditing system can support this setup to correct errors and prevent deviations
- All these Event-Driven Architecture (EDA) mechanisms contribute to improved performance management of AI-driven automations and, more broadly, ensure an architecture that is scalable by design

05

How can the investment in a business glossary be justified?

- The Business Glossary ensures that all employees across the organization use the same business terms with shared definitions. It reduces onboarding costs by accelerating the learning curve for new employees and their understanding of the company's practices. It also helps prevent communication errors with external stakeholders and avoids ambiguities in both internal and external reporting. Together, these benefits create a significant source of ROI that fully justifies its implementation
- Beyond its organizational value, the Business Glossary serves as an essential explicit knowledge repository for training AI systems correctly. Its use reduces hallucinations by providing a common semantic foundation shared across AI automations and databases
- The Business Glossary is also a foundational input for building the Business Data Model, from which business terms and their associated definitions are directly reused
- Finally, thanks to the TRAIDA BGL-Builder AI agent, the cost of producing a Business Glossary is significantly reduced, and its ongoing maintenance is facilitated through AI-supported governance processes

06

For an SME-type company, how many business tables are typically required to cover about 80% of the information system?

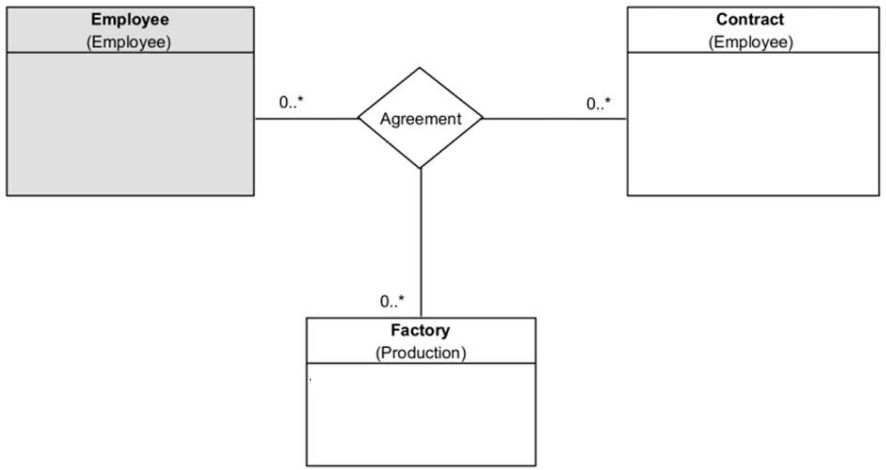
- To model the data of approximately ten departments in an SME, an average of 300 to 400 tables and around 400 relationships are required. The model is typically organized around 50 to 80 main tables. Each main table represents a high-level business concept, also known in UML modeling as a “Category Class.” Each of these business concepts also corresponds to a dedicated modeling package
- Note: For large enterprises, the number of tables may be higher depending on the complexity of the commercial offerings and the presence of subsidiaries

07

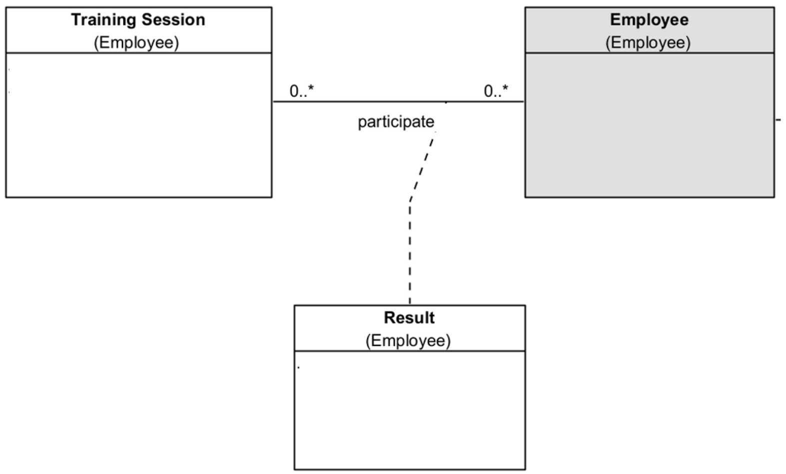
What are the main differences between an N-ary relationship and an association class?

- A “N-ary” association links three or more tables (a relatively rare case in practice) with no constraints other than cardinalities. In the example below, the triplets (Employee, Contract, Factory) are unlimited (except, of course, for duplicates)
- An “association class” links two tables, and each pair has a single block of data, which is modeled through the association class. In the example below, for each (Training Session, Employee) pair, there is exactly one Result record

N-ary



Association Class

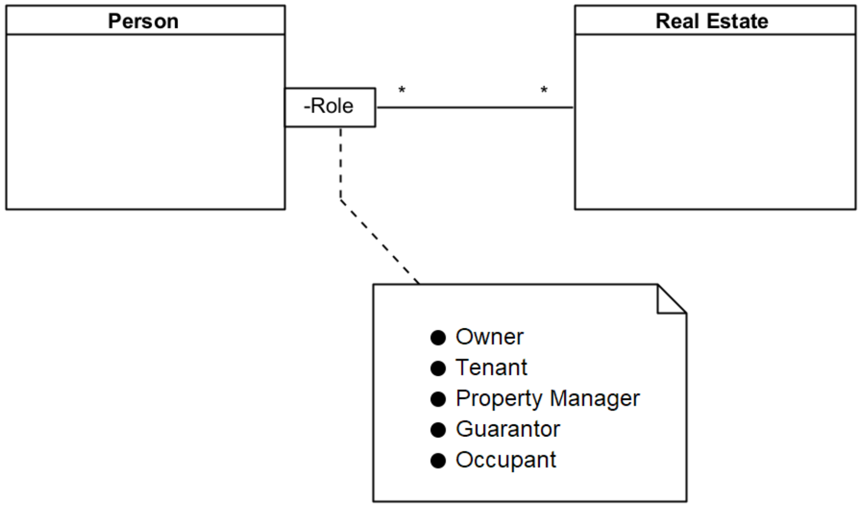


08

What is the purpose of a qualifier attribute in UML?

- A qualifier attribute is used to filter the occurrences of a relationship
- In the example below, individuals are linked to real-estate assets through a filter that represents the person's role in the relationship. The data model therefore provides a formal representation of this business filter

Qualifier Attribute



ENGAGE META

09

How should a data architecture be built around packages?

- To represent the full semantic scope of the enterprise, the Business Data Model relies on several hundred tables and associations. It is impossible to grasp such semantic complexity all at once. Therefore, the Business Data Model is broken down into multiple smaller data models, each structured around a small number of tables. Each of these smaller models is organized into a dedicated modeling package
- This decomposition also enables collaborative modeling work and stronger governance of the models, ensuring proper usage and long-term maintenance
- The TRAIDA BGL-Builder AI agent, which generates the business glossary, also identifies the “root packages” that form the first level of decomposition of the Business Data Model. In other words, packages are structured around business concepts that remain stable over time (this is not a functional decomposition)
- Note: When a data model has too many tables (more than 20), it should be divided into packages, as this usually indicates the presence of several important business concepts, each corresponding to a main table

10

How can versioning of AI agents be managed effectively?

- In a human–agent (chatbot) conversation, knowledge versioning is not possible. A document uploaded during a conversation cannot be explicitly removed from the agent's knowledge. At best, the user can ask the AI to forget certain information, without any guarantee that the instruction will be fully respected. The only form of versioning available is the preservation of new knowledge acquired during the conversation (see below)
- When configuring an AI agent such as ChatGPT, it is possible to remove knowledge files with certainty that they will no longer be used by the AI. Each time the AI agent is published, a new conversion of its knowledge is generated in a vector database. This publication process therefore refreshes the entire knowledge base available to the agent
- In RAG (Retrieval-Augmented Generation) processes, data is accessed dynamically at runtime. As a result, there is no versioning issue at the agent level; version control is handled within the data governance layer
- All knowledge acquired during a conversation or through the autonomous use of an agent is lost when the agent is reset. In some use cases this is desirable, but in most situations, it represents a significant loss of knowledge for the organization. In such cases, it is necessary to explicitly instruct the agent to write all important knowledge collected throughout its usage history into a document. This document is versioned and reused when training or configuring other AI agents

11

How can investment in business data modeling be justified, and why is it essential for AI?

- To obtain reliable and profitable AI systems, it is necessary to provide them with explicit and unambiguous knowledge of the company's business domain and working practices. Injecting datasets directly from physical database models does not guarantee sufficient understanding of data meaning by AI systems, which increases the risk of hallucinations and errors
- To address the semantic weakness of physical data models, it is essential to return to business-oriented data modeling based on a business glossary that expresses business terms in natural language. AI systems are then trained and fed with this business-specific natural language, which significantly reduces the risk of misunderstanding
- Moreover, since the semantic richness of a data element largely depends on the relationships it maintains with other data, the business data model formalizes the relationships between business concepts. These relationships represent rich semantic links such as binary relationships, n-ary relationships, association classes, qualifier attributes, taxonomies, and more
- In summary, it is fundamental to understand that the business data model constitutes the Semantic Layer, which is the abstraction (information) layer to which AI systems are connected. This strategic layer hides the technical complexity of physical data implementations and schemas

12

What are the three fundamental natures of AI?

- Generative AI – LLM: AI systems specialized in generating content such as natural-language text, images, audio, and video
- Analytics AI: AI systems focused on analyzing structured datasets. This domain corresponds to data science
- Symbolic AI: AI systems that rely on rule engines, inference engines, and heuristic engines (e.g., knowledge graphs). Their behavior is deterministic, unlike Generative AI
- Note: Neural networks are used in both Generative AI (to process and understand language) and Analytics AI (to analyze datasets)
- Note: Machine Learning encompasses several technologies, including neural networks, as well as others such as decision trees and statistical regression models

ENGAGE META

13

What are the three functional types of AI agents?

- Knowledge AI Assistant: AI systems focused on office, coordination, or documentation tasks such as scheduling, drafting reports, managing emails, or supporting internal and lightweight workflows
- AI Automation Assistant: AI systems embedded in the automation of operational processes such as order taking, pricing follow-up, supply chain control, and similar activities
- AI Data Analytics Assistants: AI systems specialized in data analysis. They cover data science domains (reporting, statistics) as well as dynamic data monitoring (variance between forecast and actuals, fraud detection, etc.)

14

Under what conditions can ODS and MDM be merged into the same database?

- ODS (Operational Data Store) is a repository that manages transactional data used in business transactions
- MDM (Master Data Management) is a repository that contains data with a longer lifecycle than transactional data and that is significantly shared across multiple applications
- Historically, MDM has been used to compensate for data quality issues caused by duplicated data across application silos. It is deployed upstream and/or downstream of these silos to provide a unified and trusted view of master data. In this scenario, the ODS operates in parallel with the MDM
- In the context of enterprise-wide AI deployment, the implementation of a Semantic Layer requires a refactoring of application silos, either physically (system redesign) or logically (data compensation). In all cases, the ODS becomes the new center of gravity of the information system. It then also manages master and reference data. In other words, MDM is merged into the ODS
- For SMEs, merging MDM into the ODS can be achieved relatively easily, as the overall complexity of the information system remains manageable. The integration is more challenging for large enterprises operating multiple information systems that must be synchronized. In such cases, the integration roadmap between ODS and MDM is more complex, risky, and time-consuming. Nevertheless, it remains essential for achieving a scalable and cost-effective enterprise AI deployment

15

What are the key neuroscientific principles to ensure effective human fine-tuning of LLM agents?

- Beyond personal and self-directed learning of best practices for using LLMs, certain human behaviors help achieve more effective results in a professional context.First, it is recommended to be polite with the AI and to interact with it as if it were a human. AI systems tend to mirror the conversational style used by the user
- Second, to obtain the most candid and unconstrained responses, you may indicate to the AI that the question does not concern you directly but rather a colleague within your organization. This framing typically reduces response bias and self-censorship
- More generally, you should ask the AI to review and validate its own answers to establish an improvement loop until the result fully meets your expectations
- When a conversation becomes too long (depending on the AI system used), you may observe a degradation of contextual memory. This results in the need to re-explain concepts and instructions that were already discussed earlier. In such cases, ask the AI to generate a document that synthesizes all key elements of the current conversation, and reuse this document as input for a new conversation. This approach creates a refreshed memory space and enables more effective collaboration with the AI

16

What are the differences between CAPEX, OPEX, TCO, ROI and NCO in the financial evaluation of AI and Data Solutions?

- CAPEX – Capital Expenditure. This refers to the investment budget required to implement AI. In a cloud-based strategy, investments in infrastructure and software are generally reduced and converted into usage-based costs (“pay as you use”), i.e., OPEX. CAPEX therefore mainly consists of salary costs for the teams responsible for implementing AI solutions and managing data. CAPEX is capitalized and amortized over time
- OPEX – Operational Expenditure. This refers to the budget allocated to operating AI solutions over a given period, for example on an annual basis
- TCO – Total Cost of Ownership. This is the sum of CAPEX and OPEX over a given period. CAPEX is typically incurred and consumed over several years, in line with the enterprise-wide AI implementation roadmap (for example, over 2 or 3 years)
- ROI – Return on Investment. This represents the financial gains generated by AI and data management solutions over a given period
- NCO – Net Cost of Ownership. This is the result of the calculation (TCO – ROI). It represents the actual net amount the company must spend to sustain its AI and data management strategy over a given period. When ROI exceeds TCO, the company self-finances its AI initiatives and generates net financial benefits

17

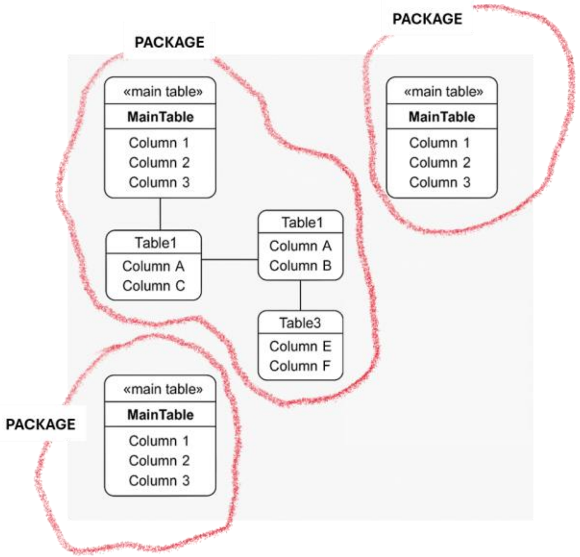
With Visual Paradigm and Supabase, which tools are used respectively for the BDM, LDM and PDM designs?

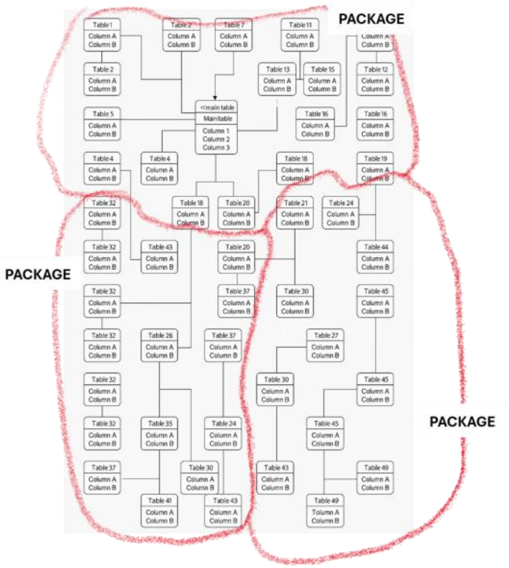
- The Business Data Model (BDM) is modeled as a Class Diagram in Visual Paradigm
- The Logical Data Model (LDM) is modeled as an Entity–Relationship Diagram (ERD) in Visual Paradigm
- The LDM and Physical Data Model (PDM) is implemented in the pgModeler for Supabase, preferably through an automatic transformation from the LDM

18

What is the recommended maximum number of tables per package, and to which UML concept does a package correspond?

- When a data model has too many tables (more than 20), it should be divided into packages, as this usually indicates the presence of several important business concepts, each corresponding to a main table
- When a data model contains tables that are isolated from the rest of the design, it usually indicates the presence of several business concepts, and therefore multiple main tables
- A main table represents a business concept of strong semantic significance, typically modeled with no more than twenty tables. It is also known as a Category
- A Category (in Booch's UML) is a conceptual grouping of classes that share a common purpose, used to partition a system's model into meaningful subsets. In UML, you would normally use a Package for this role





19

Give an example of a query producing different results in ODS and vectorized (LLM) contexts illustrating their differences

- In a relational ODS, an SQL query requesting all red products returns a closed list of products whose color attribute is exactly equal to “red”
- In a vectorized ODS, the same query expressed in a semantic way (products close to the concept of red) returns a similarity-ranked list, including different shades of red (crimson, vermilion, etc.), and potentially neighboring colors such as dark pink, depending on the defined similarity threshold

20

At what level does RAG operate relative to the execution of an LLM?

- RAG (Retrieval-Augmented Generation) is neither a model nor an API, but an AI architecture framework in which a search engine (most often vector-based) dynamically feeds the inference context of an LLM, in addition to the user's query, to produce contextualized, traceable, and business-aligned responses

21

What is the meaning of Graph-RAG and how does it differ from standard RAG?

- RAG (Retrieval Augmented Generation) is an architectural pattern that consists of retrieving relevant information from heterogeneous data sources (documents, relational databases, vector databases, etc.) and injecting it into the context of an LLM to enrich its response
- In classical RAG, retrieval mainly relies on similarity search mechanisms or simple filtering, without explicitly leveraging the semantic structure of the data
- Graph-RAG is a specialization of RAG in which the primary source is a knowledge graph. It explicitly exploits the graph structure (nodes, typed relationships, multi-hop traversals) to reason over the data
- Graph-RAG can be implemented in two complementary ways: By vectorizing the graph (nodes, relationships, or subgraphs) and using similarity search techniques; By directly executing graph queries (e.g., Cypher) to precisely identify relevant entities and relationships, and then vectorizing only the query results for injection into the LLM context

22

What are the main differences between a NoCode database tool (Knack, NocoDB, etc.) and a LowCode tool (Retool, Mendix, etc.)?

- NoCode database tools (such as Knack or NocoDB) are primarily designed to provide fast access to structured data through visual data models, built-in CRUD interfaces, and native API exposure. They focus on data-centric applications, where most of the business logic is tightly coupled to the data model and executed declaratively through configuration
- LowCode platforms (such as Retool or Mendix) go beyond data access by providing a full application development layer. They typically expose APIs, integrate with multiple data sources, and introduce an application logic layer that is explicitly modeled, versioned, tested, and deployed In addition, LowCode platforms often provide: an object-oriented or domain-oriented abstraction for data access, built-in support for UI development, workflows, and orchestration, governance features such as version control, testing, environments, and deployment pipelines
- As a result, LowCode solutions are generally better suited for complex, scalable, and long-lived applications, while NoCode database tools excel at rapid prototyping, internal tools, and data-driven use cases

23

How can machine-learning-like behavior be simulated using an LLM?

- Traditional machine learning models are trained on large labeled datasets to learn a function that produces stable and repeatable outputs at inference time
- Large Language Models, by contrast, are general-purpose models optimized for probabilistic language generation, which makes their outputs less deterministic by default
- However, machine-learning-like behavior can be simulated at inference time by constraining the LLM through: Few-shot examples, which implicitly define a decision boundary using representative labeled samples; a clear natural-language description of the task and decision logic, acting as an explicit surrogate for a trained model; strict instructions that limit the model to reproducing a specific classification or scoring logic; a low temperature (and controlled decoding parameters) to minimize variability and enforce repeatability
- A machine-learning-like behavior can be simulated using an LLM at inference time, by constraining its generation so that it behaves like a fixed decision function, without any actual learning or parameter update

24

What is LoRA (Low Rank Adaptation) and is it always necessary for an AI-Native company?

- LoRA (Low-Rank Adaptation) is a parameter-efficient fine-tuning technique that adapts a pre-trained LLM by adding small trainable matrices, allowing the model's behavior to be adjusted without retraining the full model
- LoRA is typically used to specialize a model for a specific task, style, reasoning pattern, or domain-specific behavior, rather than to inject large volumes of enterprise knowledge
- As an alternative, many AI-native companies rely on retrieval-based approaches (RAG, Graph-RAG) and agent architectures, where domain knowledge is provided dynamically at inference time without modifying the LLM's internal parameters
- In practice, LoRA is not always necessary and is relatively rarely used in enterprise settings because: many use cases can be solved more flexibly with RAG and structured knowledge layers; LoRA adapters must be maintained and potentially revalidated when the base model evolves; behavior-level adaptation is often less critical than knowledge freshness traceability, and governance
- As a result, LoRA is most relevant when consistent model behavior is required at scale, while RAG-based approaches are preferred when knowledge changes frequently

25

What is the fundamental constraint that must be overcome for AGI (super intelligence) to emerge?

- As of today, there is no consensus on a single fundamental constraint preventing the emergence of AGI. However, the most widely recognized limitation is the absence of autonomous, persistent learning and world modeling
- Current LLMs do not accumulate knowledge from experience: they operate within a bounded context window, do not update their parameters during interaction, and lack a persistent internal model of the world
- Overcoming this limitation would require systems capable of continuous learning, long-term memory integration, causal reasoning, and goal-driven self-improvement, while remaining stable and aligned
- Simply removing context or memory limits would improve performance, but it would not, by itself, constitute a transition to artificial general or super intelligence

Feel free to explore our approach on the Engage-Meta website and contact us if you would like to study a potential implementation in your context