2/26/2024

Databases

Database Design

Relational Database
Data is entered into tables, and those tables are linked through "relations". Use unique IDs called "keys".
Every table has a primary key for every entry (every row) in the table
Similar to a Lookup Table in a spreadsheet (relations)
Database administrator manages the physical storage, not the logical structure of the database
Designed to solve problems like compatibility, maintenance and to optimize performance
Typically, you access a relational database using SQL – Structured Query Language
(in contrast NoSQL is for unstructured databases)

Data consistency
NoSQL can only provide "eventual consistency" – it needs time to catch up

Commitment and atomicity—
Commitment is making a change permanent
Atomicity is there to ensure accuracy, multifaceted commitment capability

Storage procedures
Databases can have issues with concurrency and multiple users
Locking entries and establishing priorities in terms of editing

Factors
What are your data accuracy requirements?
Is the database scalable?
Is concurrency important?
Performance and reliability?

Graph DBMS – network database (Twitter)

Object-oriented databases (ODMBS)
Data is stored in objects, each object is an instance of class
Based on object structure, object classes and object identity

Object structure: properties that make up an object – attributes
Include: messages, methods, variables
Messages – communicate with the outside world
    Read only or update
Methods – return a value as an output (read only or update)
Variables – stores the value of data in an object

Advantages:
Objects are persistent
Faster database access and performance

Drawbacks:
Not as popular, so it can be hard to find developers
Not as many languages support object-oriented databases
Does not have a standard query language
Can be difficult to learn for non-programmers

Object-relational database : PostgreSQL (most popular)

Some examples of ODBMS : Cache, Concept Base, Db40, ObjectDB, Object Database, Object Store, Objectivity, Versant, WakandaDB, some popular GIS products

Archive site has a link to several SQL tutorials

JSON is document database, a kind of non-relational database
Stores data in plain text, stores queries as JSON documents
Uses a document-model format that developers use in their application code – easier to store and query data

Flexible and semi-structured
Allows the database to evolve and change

JSON (Javascript Object Notation) is built on a collection of name/value pairs – like a Python dictionary
"firstname": "Bobbie"
Values can be a string, a number, a Boolean, or object or array (list or matrix)
Can have nested structures: useful for spatial data

XML – is modeled on the structure of HTML
Pairs of tags: start tag and an end tag, and data is in between
<firstname>Bobbie</firstname>
XML – extensible markup language (behind most Office documents)
Customizable, with strict semantics
Provided a technology to store, communicate and validate any kind of data that can be easily read and processed by humans (human-readable)

AJAX – Asynchronous Javascript and XML
Web technology that communicated with background servers in Javascript w/o reloading the HTML page every time they communicated with the servers

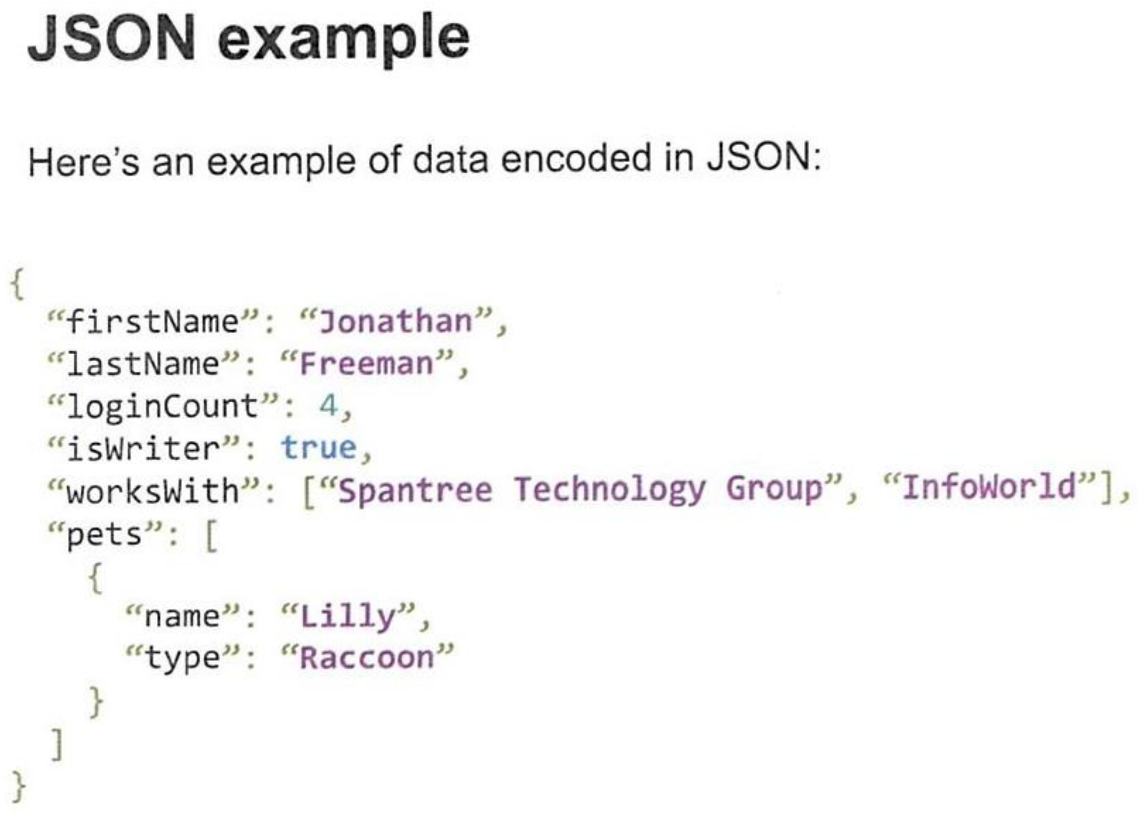
It uses HTML and CSS for presentation
Document-object model for dynamic display/data interactivity
XML for data interchange
XMLHTTP Request object for asynchronous communication with servers
Javascript knits it all together

JSON was born where AJAX was new and support for AJAX in browsers was poor
JSON was built on Javascript which browsers did support

Closing tags in XML makes the documents require more memory when stored compared to a similar JSON document

More databases have support for JSON: PostgreSQL, MySQL, MongoDB, etc.

JSON:

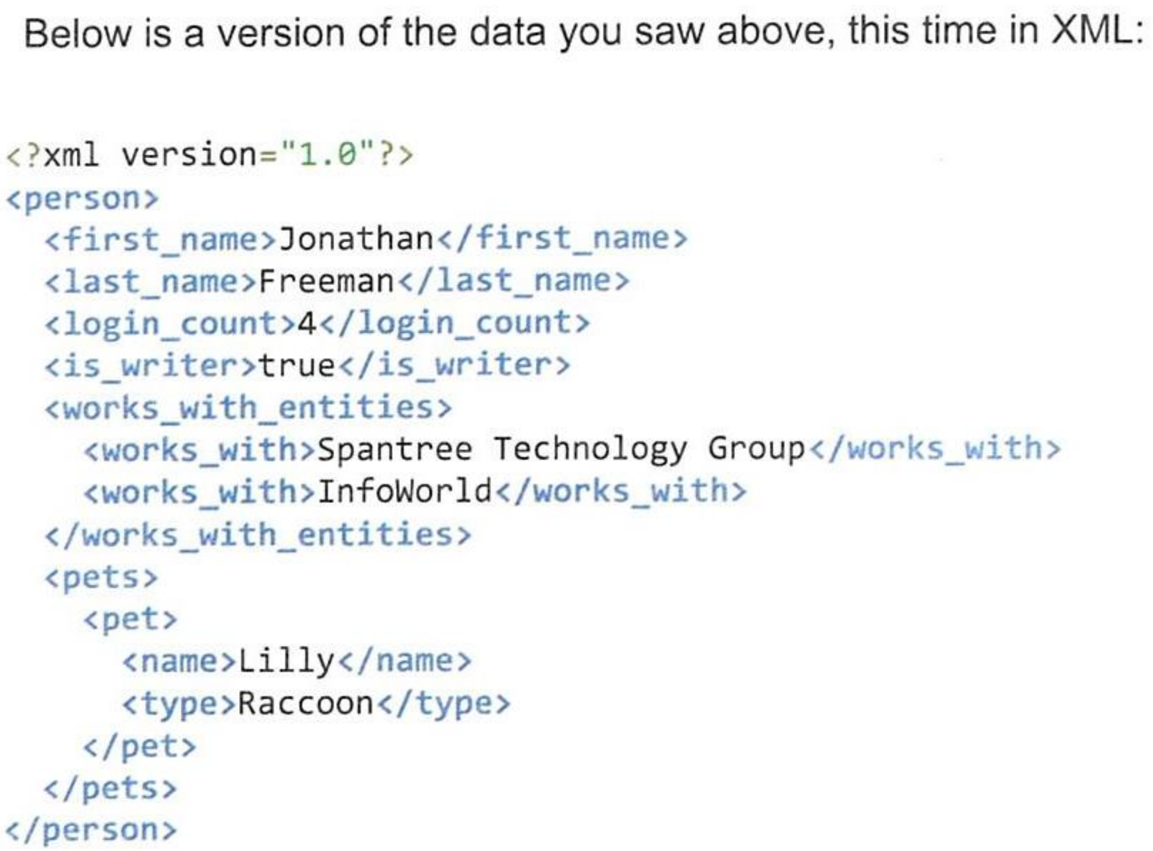


JSON example

Here’s an example of data encoded in JSON:

```
{
  “firstName”: “Jonathan”,
  “lastName”: “Freeman”,
  “loginCount”: 4,
  “isWriter”: true,
  “worksWith”: [“Spantree Technology Group”, “InfoWorld”],
  “pets”: [
    {
      “name”: “Lilly”,
      “type”: “Raccoon”
    }
  ]
}
```

XML:



Below is a version of the data you saw above, this time in XML:

```
<?xml version="1.0"?>
<person>
  <first_name>Jonathan</first_name>
  <last_name>Freeman</last_name>
  <login_count>4</login_count>
  <is_writer>true</is_writer>
  <works_with_entities>
    <works_with>Spantree Technology Group</works_with>
    <works_with>InfoWorld</works_with>
  </works_with_entities>
  <pets>
    <pet>
      <name>Lilly</name>
      <type>Raccoon</type>
    </pet>
  </pets>
</person>
```

Limitations on JSON:

- No fixed schema: flexible but can create misshapen data
- Only one number format (double precision floating point)
- No datatypes – all strings representations of data – especially for dates
- No commenting – no in-line annotations, additional documentation
- Verbosity

JSON.org is the website and it provides a list of parsers for languages to read JSON (including Python)

JSON can be converted to other file formats
ConvertCSV.com to convert to csv files, which can then be opened in Excel

Maintenance and Management of Data
Depend somewhat on the structure of the database

Data cleansing and data maintenance – for data improvement

Cleansing – tackles errors in a database, ensures retrospective anomalies are located and removed
May be done periodically (on a regular or semi-regular schedule), but infrequently

Maintenance – ongoing correction and verification – happening all the time
Continual improvement and regular checks

Software assists with both tasks

Maintenance tips:

- Keep all data in one central file or program
- Use clear descriptive names
- Add new information to the database directly
- Keep data up to date, handle changes as they occur
- Allow people/users to edit their own data
- Check permissions/take steps to prevent spam, phishing, etc. and other hacking attempts

Data management:
Acquiring data, validating data, storing data, protecting data, processing required data, ensuring accessibility, data reliability, timeliness

What kind of questions do you want to answer (about your organization/from the data)?
Correct data management results in better data analysis: do you have the right data? The right tools to tell the story?

Data governance – planning aspects
Data architecture – structure of the data
Data modeling and design – analysis
Data storage and operations – physical hardware
Data security – protecting the data
Data integration and interoperability – how well is the data integrated into your organization and other tools
Data documents and content – unstructured
Reference and master data
Datawarehousing and Business intelligence
Metadata – data about the data
Data quality

Resources:

1. https://www.oracle.com/database/what-is-a-relational-database
2. https://www.geeksforgeeks.org/definition-and-overview-of-odbms/
3. https://www.c-sharpcorner.com/article/what-are-object-oriented-databases-and-their-advantages2/
4. https://aws.amazon.com/documentdb/what-is-json
5. https://www.json.org/json-en.html
6. https://www.toptal.com/web/json-vs-xml-part-1
7. https://www.infoworld.com/article/3222851/what-is-json-a-better-format-for-data-exchange.html
8. https://medium.com/@robertopreste/from-xml-to-pandas-dataframes-9292980b1c1c
9. https://tdan.com/data-cleansing-vs-data-maintenance-which-one-is-most-important/19157
10. https://www.copernica.com/en/documentation/database-maintenance
11. https://ngdata.com/what-is-data-management
12. https://www.sas.com/en_us/insights/data-management/data-management.html
13. https://www.techrepublic.com/article/data-management-a-cheat-sheet/

Extended commentary:

Database design is the process of designing the structure of a database system, including the tables, columns, relationships, and constraints that define how data is organized and stored. A well-designed database is crucial for ensuring efficient and accurate data storage, retrieval, and manipulation.

The process of database design typically includes the following steps:

Requirements gathering: Determining the needs of the users and the data that needs to be stored in the database.

Conceptual design: Creating an initial model of the database, including the entities and their relationships.

Logical design: Creating a detailed model of the database, including the tables, columns, and relationships.

Physical design: Defining the physical characteristics of the database, such as the storage requirements and indexing strategies.

Implementation: Creating the database schema and loading the data.

Testing and optimization: Validating the database design and optimizing performance as needed.

Some key considerations in database design include:

Data integrity: Ensuring that data is accurate and consistent across the database.

Scalability: Designing the database to handle increasing amounts of data and users over time.

Performance: Optimizing the database design to ensure fast retrieval and manipulation of data.

Security: Ensuring that data is protected from unauthorized access and that sensitive data is encrypted or masked as needed.

Maintainability: Creating a design that is easy to maintain and update over time.

Usability: Designing the database with the needs of the users in mind, including ease of use and accessibility.

Overall, database design is a critical component of any data management system, and a well-designed database can ensure efficient and effective data storage and retrieval for a wide range of applications.

SQL (Structured Query Language) is a programming language used to manage and manipulate relational databases. A relational database is a type of database that organizes data into one or more tables, with each table consisting of rows and columns. Each table represents a type of entity, and each row represents a single instance of that entity.

SQL is used to create, modify, and query data stored in a relational database. Some common SQL commands include:

SELECT: retrieves data from one or more tables

INSERT: adds new data to a table

UPDATE: modifies existing data in a table

DELETE: removes data from a table

CREATE: creates a new table, view, or other database object

ALTER: modifies the structure of an existing table or other object

DROP: deletes an existing table or other object

Relational databases have several advantages over other types of databases, including:

Data integrity: relational databases enforce rules to ensure that data is consistent and accurate, preventing errors and inconsistencies.

Scalability: relational databases can handle large amounts of data and support multiple users accessing the data simultaneously.

Flexibility: relational databases can be easily modified to accommodate changes to the data or business requirements.

Security: relational databases can implement security measures to protect data from unauthorized access.

However, relational databases also have some disadvantages, including:

Complexity: designing and maintaining a relational database can be complex, requiring specialized skills and knowledge.

Limited performance: relational databases can be slow when dealing with large amounts of data or complex queries.

Data redundancy: relational databases can suffer from data redundancy, with the same data stored in multiple tables, leading to inefficiencies and potential data inconsistencies.

Overall, SQL and relational databases are widely used in the industry due to their flexibility, data integrity, and scalability.

Besides relational databases, there are several other kinds of databases. Some of them are:

NoSQL databases: These databases don't use a tabular schema like relational databases. Instead, they use various data models like document, key-value, graph, or column-family. Examples of NoSQL databases include MongoDB, Cassandra, and Redis.

Object-oriented databases: These databases are designed to store object-oriented data. They use object-oriented concepts like classes, inheritance, and polymorphism to manage the data. Examples of object-oriented databases include db4o and ObjectStore.

Hierarchical databases: These databases organize data in a tree-like structure where each record has a parent-child relationship. The parent record can have many child records, but each child record can have only one parent. Examples of hierarchical databases include IBM's Information Management System (IMS) and Windows Registry.

Graph databases: These databases use graph structures to store data, where each node represents an entity, and each edge represents a relationship between two nodes. Graph databases are useful for analyzing complex, interconnected data. Examples of graph databases include Neo4j and OrientDB.

Document databases: These databases store semi-structured or unstructured data as documents. Each document contains key-value pairs, and the documents can be organized into collections. Examples of document databases include MongoDB and Couchbase.

Each type of database has its own strengths and weaknesses, and the choice of database depends on the specific use case and data requirements of the application.

Relational databases and object-oriented databases are two different types of databases with their own set of advantages and disadvantages.

Advantages of Relational Databases:

The data in relational databases is well-organized and structured, which makes it easy to maintain and query.

It enforces data integrity and consistency by allowing the use of constraints and relations.

It provides support for transactions, which ensures that the data is consistent and accurate.

Disadvantages of Relational Databases:

Scaling can be a challenge as the structure and relationships are pre-defined.

It can be difficult to handle complex data structures, such as hierarchical or graph-based data.

It can be rigid in its handling of data, which can make it difficult to adapt to changing data needs.

Advantages of Object-Oriented Databases:

It can handle complex data structures such as hierarchical or graph-based data.

It can be more flexible in handling data, which makes it easier to adapt to changing data needs.

It can be more efficient in handling certain types of data, such as multimedia or scientific data.

Disadvantages of Object-Oriented Databases:

It can be challenging to integrate with existing systems that use relational databases.

It can be less mature and less well-established than relational databases, which can make it harder to find support or expertise.

It can be more complex to manage and maintain, which can increase costs and decrease efficiency.

JSON (JavaScript Object Notation) and XML (Extensible Markup Language) are both widely used for data interchange in modern software systems. Here are some key similarities and differences between the two:

Similarities:

Both are text-based formats used for data interchange.

Both can be used to represent hierarchical data structures.

Both are human-readable and can be parsed by machines.

Differences:

JSON is more lightweight than XML and is easier to read and write.

JSON is better suited for representing structured data, such as arrays and key-value pairs. XML is more flexible and can represent data with more complex structures and relationships.

JSON is typically used for client-server communication, while XML is often used for data storage and retrieval.

JSON has better support for arrays, making it easier to manipulate array-based data structures. In contrast, XML requires more complex parsing to work with array-based data.

JSON supports fewer data types than XML, which can be an advantage or disadvantage depending on the specific use case.

Overall, JSON is often preferred for its simplicity and ease of use, while XML is better suited for complex data structures and use cases where flexibility is important.

Maintenance and management of data involve activities and processes that ensure data is reliable, accurate, and accessible over time. These processes include data cleaning, data integration, data transformation, backup and recovery, security, and performance tuning. Effective maintenance and management of data ensure that data remains useful and relevant to users, and that it can be retrieved and utilized for analysis or decision-making purposes.

Data cleaning involves detecting and correcting or removing inaccuracies, inconsistencies, or incomplete records in the data. Data integration involves combining data from multiple sources into a single view, enabling data analysis and decision making. Data transformation involves changing the format, structure, or values of data, to enable integration and analysis.

Backup and recovery involve creating copies of data and storing them in secure locations to ensure that data can be recovered in case of system failures, disasters, or data breaches. Security involves protecting data from unauthorized access, ensuring data privacy, and preventing data breaches. Performance tuning involves optimizing data access and retrieval, ensuring that data can be accessed efficiently and quickly.

Effective maintenance and management of data require the use of tools and technologies such as database management systems, data warehousing, and big data technologies. It also requires the establishment of data governance policies and procedures that ensure data quality, integrity, and security. Effective data management practices also require the involvement of data stewards, data owners, and data custodians who are responsible for managing and maintaining data throughout its lifecycle.

Data governance refers to the process of managing the availability, usability, integrity, and security of the data used in an organization. It is an overall framework that outlines the roles, responsibilities, and processes involved in managing data. The goal of data governance is to ensure that data is accurate, complete, and reliable and that it is used appropriately and securely.

Data governance involves the establishment of policies, procedures, and standards for managing data. It also involves the implementation of technologies, processes, and controls to ensure that data is used and managed effectively. The data governance process includes the following steps:

Data discovery and classification: The identification of data sources, types, and sensitivity levels.

Data ownership and accountability: The assignment of ownership and accountability for data management.

Data quality management: The establishment of processes and standards to ensure the accuracy, completeness, and consistency of data.

Data security and privacy: The implementation of processes and controls to protect data from unauthorized access, theft, and misuse.

Data lifecycle management: The management of data throughout its lifecycle, from creation to archiving and disposal.

Compliance management: The implementation of policies and procedures to ensure that data management practices comply with regulatory requirements.

Effective data governance can improve the quality and consistency of data, reduce the risk of data breaches and noncompliance, and increase the value of data to the organization. It requires collaboration and communication across departments and stakeholders to ensure that data is managed effectively and in line with organizational goals and objectives.

Data architecture refers to the overall design and structure of an organization's data environment. It includes the processes, technologies, policies, standards, and models that govern the collection, storage, integration, management, and use of data. The goal of data architecture is to ensure that an organization's data is accurate, consistent, secure, and easily accessible to support business objectives.

Data architecture involves several key components, including:

Data models: A data model is a visual representation of the data structure, which describes the entities, attributes, and relationships between data elements. Data models help to ensure data consistency and accuracy across different systems.

Data integration: Data integration involves combining data from different sources to provide a unified view of the data. This is typically done using ETL (extract, transform, load) tools or APIs.

Data storage: Data storage refers to the physical location where data is stored. This can include traditional relational databases, NoSQL databases, data warehouses, and data lakes.

Data governance: Data governance involves establishing policies, standards, and procedures to ensure the proper use, management, and security of an organization's data.

Data security: Data security involves protecting data from unauthorized access, theft, or corruption. This can include access controls, encryption, firewalls, and other security measures.

Data quality: Data quality refers to the accuracy, completeness, and consistency of data. This is critical to ensure that the data can be trusted for decision-making.

Effective data architecture can provide several benefits to an organization, including improved data quality, increased efficiency, better decision-making, and reduced costs.

Data modeling and design is the process of creating a conceptual and logical representation of data, which can be used to design a database or data warehouse. It involves identifying the entities, attributes, and relationships of the data, and creating a data model that reflects the structure and constraints of the data.

The data modeling and design process typically involves the following steps:

Requirements gathering: This involves understanding the business needs and requirements for the data, and identifying the stakeholders who will be using the data.

Conceptual modeling: This involves creating a high-level view of the data, identifying the entities, attributes, and relationships of the data.

Logical modeling: This involves creating a more detailed view of the data, and defining the structure and constraints of the data.

Physical modeling: This involves creating a detailed view of the data, and defining the storage structures, file formats, and access methods for the data.

Implementation: This involves actually creating the database or data warehouse based on the data model that was created in the previous steps.

Data modeling and design is important because it helps to ensure that the data is organized in a way that is consistent, accurate, and efficient. By creating a data model, you can ensure that the data is well-structured and can be easily accessed and analyzed. Additionally, by designing the data model in a way that meets the needs of the stakeholders, you can ensure that the data is useful and relevant to the business.

Overall, data modeling and design is a critical aspect of any data management or analytics project, as it lays the foundation for the organization, storage, and analysis of the data.

Data storage and operations are important aspects of data management, which involve the storage, retrieval, and manipulation of data. Data storage refers to the physical storage of data, including how it is organized, accessed, and backed up, while data operations refer to the processes and procedures used to manage and manipulate the data.

Data storage can be managed using various storage systems, including databases, data warehouses, data lakes, and cloud-based storage solutions. The choice of storage system depends on the type of data, its volume, velocity, and variety, and the intended use of the data.

Data operations involve various processes, such as data ingestion, data processing, data transformation, and data analysis. These processes require specific tools and technologies, such as data integration tools, data processing engines, data transformation tools, and business intelligence software.

Data storage and operations are critical components of data management, as they ensure that data is stored securely, is easily accessible, and can be analyzed and processed efficiently. Good data storage and operations practices require regular monitoring and maintenance to ensure that data is accurate, up-to-date, and accessible to authorized users. Additionally, data storage and operations must comply with relevant data privacy and security regulations.

Data integration and interoperability are critical aspects of managing and analyzing data. Data integration is the process of combining data from different sources to create a unified view of the data. Interoperability, on the other hand, refers to the ability of systems and applications to communicate and share data with each other.

Data integration involves a variety of techniques and technologies, including Extract, Transform, Load (ETL) tools, data mapping, and data warehousing. ETL tools are used to extract data from various sources, transform the data into a common format, and load the data into a target system. Data mapping is the process of creating a common data model to facilitate the integration of data from different sources.

Interoperability requires the use of common data formats, protocols, and standards to ensure that systems and applications can communicate with each other. For example, using open standards like XML, JSON, and RESTful web services can make it easier to exchange data between different systems.

The benefits of data integration and interoperability include the ability to:

Reduce data redundancy and inconsistency

Increase data accuracy and completeness

Improve data sharing and collaboration

Support decision-making processes by providing a unified view of data from multiple sources

However, there are also challenges associated with data integration and interoperability, such as:

Complexity of integrating data from multiple sources

Maintaining data quality and consistency across different systems

Ensuring security and privacy of data during integration and exchange

Managing and resolving data conflicts that may arise during integration

Overall, effective data integration and interoperability require careful planning, a clear understanding of data sources and requirements, and the use of appropriate tools and technologies to ensure that data can be effectively shared and utilized across different systems and applications.

Data documents and content refer to the various forms of documentation that accompany data throughout its lifecycle. This can include metadata, data dictionaries, data lineage and provenance information, and other documentation that describes the structure, contents, and quality of the data.

Metadata is information about the data that provides context and helps users understand its meaning and significance. It can include information such as data source, data type, date of creation, and data owner. Data dictionaries provide a more detailed description of the data's structure and content, including field names, data types, and other attributes.

Data lineage and provenance information track the history of the data from its creation to its current state. This can include information about who created the data, when and how it was transformed, and how it has been used over time. This information is important for data quality and governance, as it helps to ensure that the data is accurate and reliable.

Data documents and content are essential for data management and analysis, as they provide the context and information needed to understand and use the data effectively. They also help to ensure that the data is consistent and well-maintained over time, which is essential for data quality and governance.

Reference data and master data are two important concepts in data management. While they are related, they serve different purposes.

Reference data refers to a set of values that are used to categorize or classify data within an organization. Reference data values are typically standard, universal, and static, and they remain constant over time. Examples of reference data include product codes, geographic codes, and currency codes.

Master data, on the other hand, refers to a set of core data elements that are used across an organization to support business processes and decision-making. Master data is typically specific to an organization, and it includes data about customers, suppliers, products, and other key entities. The goal of master data management is to ensure that master data is accurate, consistent, and up-to-date across all applications and systems.

The main difference between reference data and master data is that reference data is used to categorize or classify data, while master data is used to describe key entities in an organization. Reference data is usually managed by a central authority, while master data is managed by individual departments or business units.

Both reference data and master data are critical to effective data management. Reference data helps ensure that data is consistent and accurate, while master data provides a comprehensive view of an organization's key entities. By managing reference and master data effectively, organizations can improve the quality of their data and make better-informed decisions.

Data warehousing and business intelligence (BI) are two important concepts in the field of data management and analytics. A data warehouse is a large, centralized repository of data that is designed to support business intelligence activities. It is a type of database that is optimized for querying and analysis rather than transaction processing.

Business intelligence, on the other hand, refers to the process of extracting insights from data in order to inform business decisions. This involves using various analytical tools and techniques to analyze data from multiple sources, and presenting the results in a way that is easy to understand and actionable.

The main goal of data warehousing is to provide a single source of truth for business data. By consolidating data from multiple sources into a centralized repository, organizations can avoid the problems of data silos, duplication, and inconsistency that often arise when data is stored in multiple places. This makes it easier to analyze and report on business data, and to use it to inform decision-making.

Business intelligence, meanwhile, is focused on using data to gain insights and make better decisions. This involves a variety of techniques, such as data mining, predictive analytics, and visualization, to uncover patterns and relationships in the data. BI tools allow organizations to analyze data in real-time, and to present it in a way that is meaningful and actionable.

The combination of data warehousing and business intelligence is a powerful one. By consolidating data into a single repository and using sophisticated analytical tools to analyze it, organizations can gain a deep understanding of their business and make more informed decisions. This can lead to better outcomes, increased efficiency, and a competitive advantage in the marketplace.