2/28/2022

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

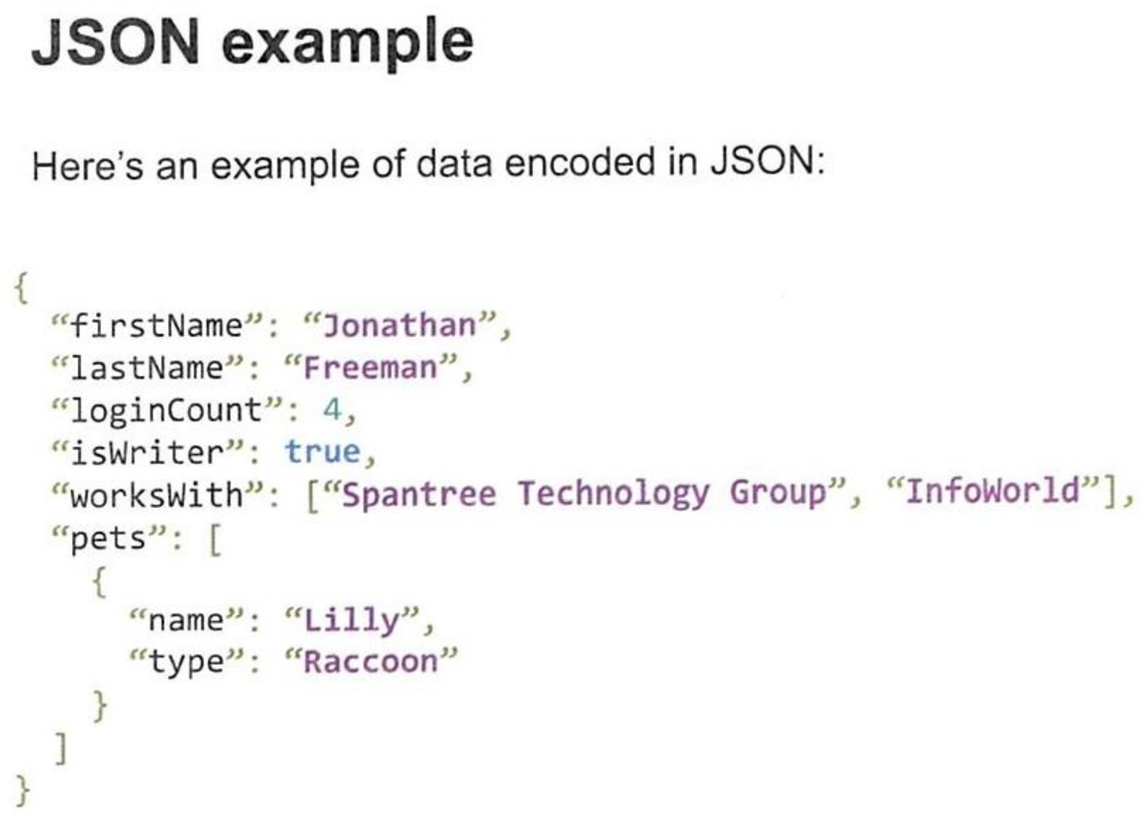
It uses HTML and CSS for presentation
Document-object model for dynamic display/data interactivity
XML for data interchange
XMLHTTP Request object for asynchronous communication with servers
Javascript knits it all together

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:

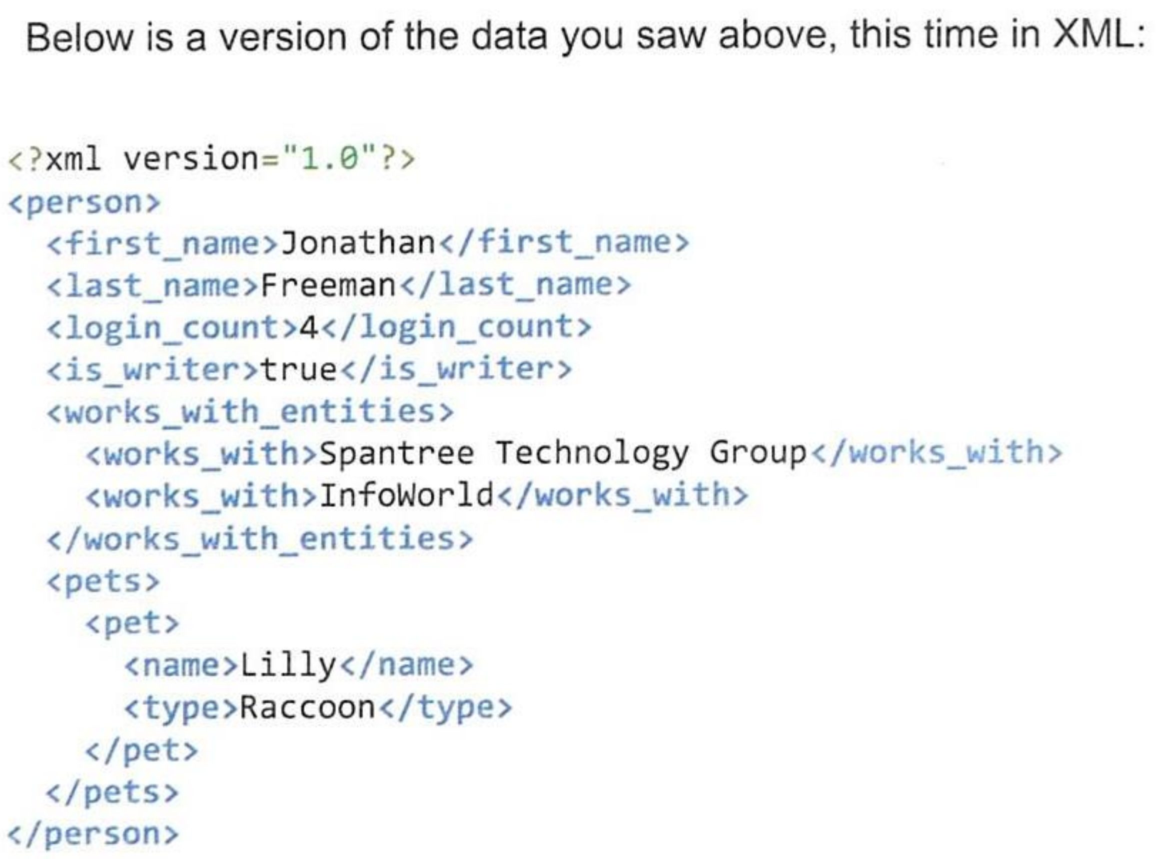


JSON example

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