3/01/2021

Databases

Database Design

Relational Database
Data is entered into tables, and those tables are linked "relations" using unique ID = key
Every table has a primary key for every entry in the table (each row)
Similar to Lookup Table
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 to NoSQL – for unstructured databases)

Data consistency
NoSQL can only provide "Eventual consistency" – time to catch up

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

Stored 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?
Concurrency important?
Performance and reliability?

Graph DBMS – network database (Twitter)

Object-Oriented Database – ODMBS
Data is stored in objects (not tables), each object is an instance of a class

Based on object structure, object classes and object identity

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

Advantages:
Objects are persistent

Faster database access and performance

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

Object-relational databases: PostgreSQL (most popular)

ODMBS: examples: Cache, Concept Base, Db40, ObjectDB, Object Database, Object Store, Objectivity, Versant, WakandaDB ; popular GIS products

On my Archive, I posted a link to a SQL cheat sheet

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

Flexible, semi-structured
Allows the database to evolve and change is needed

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

XML – is modeled on the structure of HTML
Pairs of tags: start tag, end tag, and data 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 (human-readable)

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

--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 to knit it all together

JSON (Javascript Object Notation) 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 make the documents more memory intensive when stored compared to an equivalent JSON file

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

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"
    }
  ]
}
```

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 of JSON

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

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

JSON can be converted to other data format files:
ConvertCSV.com – to CSV files, which can then be opened in Excel

Maintenance and Management of Data
Depend somewhat structure of database

Data cleansing and data maintenance – for data improvement

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

Maintenance – ongoing correction and verification – happening all the time
Continual improvement, regular checks
Software exists assist 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
Impacts several other areas:
Such as data analysis, data collection

What kind of questions do you want to answer?
Correct data management results in better 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 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 BI
Metadata – data about the data
Data quality