Topic 16:

An Introduction to ‘NoSQL’ DBMSes

Relational DBMSes: A Review

Hopefully mostly old news!

- Schemas describe highly–structured data
- Originally monolithic (centralized storage and control)
- ACID properties help to safeguard the data
- Designed for business data (numbers, strings)
- Performs well on transactional and analytical workloads
- Uses SQL as DDL, DML, DCL, and QL.

Relational DBMSes: Some Lingering Problems

- Challenging to Scale Up to Modern Data & Storage
- Rigid Schemas
- ACID properties limit DBMS performance

The 'Vees' of Big Data

The original three issues/characteristics of Big Data:

1.
2.
3.

Then others were added, including:

4. Veracity ('dirty' data must be cleaned)
5. Value (of what use is the data?)
6. Validity (how relevant is the data for our needs?)
7. Volatility (for how long is the data useful?)

DBMS Support for the ‘Vees’

Big–data DBMSes usually need to:

- Accept/Store/Process lots of data in real time
- Scale to larger workloads with commodity hardware
- Allow DB schemas to be flexible
- Relax enforcement of ACID properties

What is a ‘NoSQL’ DBMS?

First: ‘NoSQL’ is a terrible name for such systems!

- Why? The ideas have little to do with SQL directly
- Thus, some people now say it means ‘Not (only) SQL’; still not useful
- Maybe ‘Beyond Relational’? Branding needs work!

Five Varieties of NoSQL DBMSes (1 / 7)

1. **Attribute–Value Systems** (a.k.a., Key–Value)

   - In programming, also known as dictionaries / hash tables
   - Are *opaque* — structure of values isn't a concern
   -
   -

Five Varieties of NoSQL DBMSes (2 / 7)

2. **Wide–Column Stores**

   - A *wide–column* is a group of attribute–value pairs describing one entity
   - Unlike a relation, each group can have different collections of pairs
   - Like a relation, each group can have a row id (a.k.a., a key)
   - By contrast, a *column store* stores data by columns (often partitioned) instead of by rows

Five Varieties of NoSQL DBMSes (3 / 7)

3. **Document Systems**

- Are based on attribute–value systems, as are wide–column stores
- Typically use XML (eXtensible Markup Language), JSON (Javascript Object Notation), or BSON (Binary JSON) for data storage
- Motivating example: Your resume has your name, address, list of schools attended, current employer, past employers, etc.
  - An RDBMS would store these items in separate tables:
  -

Five Varieties of NoSQL DBMSes (4 / 7)

3. **Document Systems** (continued)

Example(s): A JSON representation of resume info:

```
{ "name" : "Amy",
  "address" : "123 Main Street",
  "degree" : [ "M.S.", "B.S." ],
  "experience" : [ { "position" : "tutor",
                     "employer" : "TUSD" },
                   { "position" : "volunteer",
                     "employer" : "Community Food Bank" }
                 ]
}
```

Five Varieties of NoSQL DBMSes (5 / 7)

4. **Graph Systems**

- Use graph data structures to store data and relationships
  - Vertices: Hold the data
  - Edges: Are the relationships
- Representative Applications:
  -
  -
  -

Five Varieties of NoSQL DBMSes (6 / 7)

5. **Vector Databases**

- Purpose: Store vectors that support LLMs
- Can be stand–alone or within another DBMS, e.g.:
  - Stand–alone:
  - Integrated:
- Include AI support algorithms, e.g.:
  -
  -

Five Varieties of NoSQL DBMSes (7 / 7)

5. **Vector Databases** (continued)
   - Many familiar features appear in Vector Databases, including:
     - Fault Tolerance
     - System Monitoring
     - Access Control
     - Data Backups
     - APIs / SDKs

Closing Remarks

- Clearly, NoSQL is a large umbrella
- Many NoSQL systems still offer:
  -
  -
- Beware the marketing hype!
  -
  -