Database Management Systems - I, CS 157A

SQL vs. NoSQL

Outline

- **SQL Databases**
  - SQL Standard
  - SQL Characteristics
  - SQL Database Examples
- **NoSQL Databases**
  - NoSQL Defintion
  - General Characteristics
  - NoSQL Database Types
  - NoSQL Database Examples
- **Summary**

SQL Databases

SQL DBMS: SQL Standard

The following is a short, incomplete history of the SQL Standards – ISO/IEC 9075

- **1987** – Initial ISO/IEC Standard
- **1989** – Referential Integrity
- **1992** – SQL2
  - **1995** SQL/CLI (ODBC)
  - **1996** SQL/PSM – Procedural Language extensions
- **1999** – User Defined Types
- **2003** – SQL/XML
- **2008** – Expansions and corrections
- **2012** – System Versioned and Application Time Period Tables

SQL DBMS: SQL Characteristics

- Data stored in rows or columns into tables (relations)
- **Relationships** represented by data
- Data adheres strictly to schema stored in the catalog.
- Complete separation between data and schema (provides flexibility)
- **Data Definition Language** (DDL – create table)
- **Data Manipulation Language** (DML - IUD)
- Complete separation between logical and physical layer/view of the DBMS. User sees only the logical view.
- **Optimizer** that allows the user to ask for the "what" and it generate code to decide the "how."
- **Transactions** – ACID properties with strong consistency

SQL DBMS: SQL Characteristics

- **Data stored in rows or columns into tables (relations):**
  - Row-oriented database is more appropriate for insert
  - Column-oriented database is more appropriate for Data Warehouse (SELECT statements)
  - Row vs. Column is pure performance issue based on the workload; the user is not aware and deals with tables.

SQL DBMS: SQL Characteristics

- **Relationships represented by data:**
  - Data in a database instance is partitioned among tables but useful operations require access to relevant data from the different tables; hence we need to be able to do JOIN.
  - Relationship between two tables is represented by primary and foreign keys attributes.

SQL DBMS: SQL Characteristics

- **Data adheres strictly to schema stored in the catalog:**
  - Data has to be 100% compliant with the schema (data types) as well as meeting integrity constraints
  - This guarantee is done before the data gets into the database (ETL).
  - Since ETL typically is semi-automatic process; hence access and process data in a SQL engine from other data sources is a challenge.
  - The data in the database is assumed to be accurate and as a result you can perform analysis and come up with reliable recommendations

SQL DBMS: SQL Characteristics

- **Complete separation between data and schema (provides flexibility):**
  - By having the schema (table definitions) away from the data in the tables, we can easily at the query level change the attribute name being presented to the user.
  - While user create table with set of attributes in some order, the database engine can change the order of attributes for performance reasons (all fixed width attributes first before variable width attributes.

SQL DBMS: SQL Characteristics

- **Data Definition Language (DDL):**
  - Schema defined at the start
  - Create Table (Column1 Datatype1, Column2 Datatype 2, …)
  - Constraints to define and enforce relationships
    - Primary Key
    - Foreign Key
    - Etc.
  - Triggers to respond to Insert, Update , & Delete
  - Stored Modules
  - Alter …
  - Drop …
  - Security and Access Control

SQL DBMS: SQL Characteristics

- **Data Manipulation Language (DML):**
  - Schema defined at Data manipulated with Select, Insert, Update, & Delete statements
    - Select T1.Column1, T2.Column2 …
From Table1, Table2 …
Where T1.Column1 = T2.Column1 …
  - Data Aggregation
  - Compound statements (sub-queries)
  - Functions and Procedures
  - Explicit transaction control

SQL DBMS: SQL Characteristics

- **Complete separation between logical and physical layer/view of the DBMS:**

  - **Physical layer** can change without any impact on the application; table grow to use two disk drives instead of one – transparent to the application

  - **Order of a table attributes** is transparent to the application; controlled by the physical layer for performance reasons.
    - Create an index to support query performance.
    - Fixed size attributes are always first before any variable width attribute.
    - In-Memory Database.

SQL DBMS: SQL Characteristics

- **Optimizer:**
  - Allows the user to ask for the "what" and the optimizer generates code to decide the "how."
  - Programming in SQL is very productive as the optimizer does most of the work that otherwise you have to do in 3rd generation programming languages.

SQL DBMS: SQL Characteristics

- **Transactions (ACID Properties):**
  - **A**tomic – All of the work in a transaction completes (commit) or none of it completes.
  - **C**onsistent – A transaction transforms the database from one consistent state to another consistent state. Consistency is defined in terms of constraints.
  - **I**solated – The results of any changes made during a transaction are not visible until the transaction has committed.
  - **D**urable – The results of a committed transaction survive failures.

SQL DBMS: SQL Database Examples

- **Commercial:**
  - IBM DB2
  - Oracle RDMS
  - Microsoft SQL Server
  - Teradata
  - HP/Vertica
  - Sybase SQL Anywhere
- **Open Source (with commercial options):**
  - MySQL
  - Ingres
- **Significant portions of the world's economy use SQL databases!**

NoSQL Databases

NoSQL Database: NoSQL Definition

- **From www.nosql-database.org:**
  - **Next Generation Databases mostly addressing some of the following points:**
    - Being **non-relational,**
    - **Distributed,**
    - **Open-source** and
    - **Horizontal scalable**.
  - The original intention has been **modern web-scale databases**.
  - The movement began early 2009 and is growing rapidly. Often more characteristics apply as: **schema-free, easy replication support, simple API, eventually consistent** / **BASE** (not ACID), a **huge data amount**, and more.

NoSQL Database: NoSQL Definition

- **NoSQL Products & Projects:**
  - http://www.nosql-database.org/ lists 122 NoSQL Databases:
    - Cassandra
    - Hive
    - Pig
    - Couchbase
    - Hadoop & Hbase
    - MongoDB
    - StupidDB
    - Etc.

NoSQL Database: General Characteristics

- **NoSQL General/Distinguishing Characteristics:**
  - Large data volumes
    - Google's "big data"
  - Scalable replication and distribution
    - Potentially thousands of machines
    - Potentially distributed around the world
  - Queries need to return answers quickly
  - Mostly query, few updates
  - Asynchronous Inserts & Updates
  - Schema-less
  - ACID transaction properties are not needed – **BASE** (Basically Available Soft state, Eventually consistent)
  - **CAP** (Consistency, Availability, Partition tolerance) Theorem
  - Open source development

NoSQL Database: General Characteristics

- **BASE Transactions:**
  - **Acronym contrived to be the opposite of ACID**
    - **B**asically **A**vailable,
    - **S**oft state,
    - **E**ventually Consistent
  - **Characteristics**
    - Weak consistency – stale data OK
    - Availability first
    - Best effort
    - Approximate answers OK
    - Aggressive (optimistic)
    - Simpler and faster

NoSQL Database: General Characteristics

Brewer's CAP Theorem:

- A distributed system can support only two of the following characteristics:
  - **C**onsistency
  - **A**vailability
  - **P**artition tolerance

P.S. The slides from Brewer's July 2000 talk do not define these characteristics.

NoSQL Database: General Characteristics

- **Consistency:**
  - all nodes see the same data at the same time – Wikipedia
  - client perceives that a set of operations has occurred all at once – Pritchett.
  - More like Atomic in ACID transaction properties.
- **Availability:**
  - Node failures do not prevent survivors from continuing to operate.
  - Every operation must terminate in an intended response.
- **Partition Tolerance:**
  - The system continues to operate despite arbitrary message loss.
  - Operations will complete, even if individual components are unavailable.

NoSQL Database: NoSQL Database Types

Discussing NoSQL databases is complicated because there are a variety of types:

- **Column Store** – Each storage block contains data from only one column
- **Document Store** – stores documents made up of tagged elements
- **Key-Value Store** – Hash table of keys

NoSQL Database: NoSQL Database Types

- **Additional Non-SQL Databases:**
  - XML Databases
  - Graph Databases
  - Codasyl Databases: data processing consortium developed COBOL + fed into ISO, ANSI for effective data systems analysis, design, and implementation.
  - Object Oriented Databases
  - Etc…

Will not address the above here..

NoSQL Database: NoSQL Examples

- **Column Store:**
  - Each storage block contains data from only one column
  - **Example:** Hadoop/Hbase
    - http://hadoop.apache.org/
    - Yahoo, Facebook
  - **Example:** Ingres VectorWise
    - Column Store integrated with an SQL database
    - http://www.ingres.com/products/vectorwise

NoSQL Database: NoSQL Examples

- **Column Store Comments:**
  - More efficient than row (or document) store if:
    - Multiple row/record/documents are inserted at the same time so updates of column blocks can be aggregated
    - Retrievals access only some of the columns in a row/record/document

NoSQL Database: NoSQL Examples

- **Document Store:**
  - **Example:** CouchDB
    - http://couchdb.apache.org/
    - BBC: multi-master multi-datacenter fail configuration
  - **Example:** MongoDB
    - http://www.mongodb.org/
    - Foursquare, Shutterfly
  - JSON – JavaScript Object Notation

NoSQL Database: NoSQL Examples

- **Key-Value Store:**
  - Hash tables of Keys
  - Values stored with Keys
  - Fast access to small data values
  - **Example** – Project-Voldemort
    - http://www.project-voldemort.com/
    - Linkedin
  - **Example** – MemCacheDB
    - http://memcachedb.org/
    - Backend storage is Berkeley-DB

NoSQL Database: NoSQL Examples

- **Map Reduce:**
  - Technique for indexing and searching large data volumes
  - Two Phases, Map and Reduce
    - **Map**:
      - Extract sets of Key-Value pairs from underlying data
      - Potentially in Parallel on multiple machines
    - **Reduce**:
      - Merge and sort sets of Key-Value pairs
      - Results may be useful for other searches
  - Map Reduce techniques differ across products
  - Implemented by application developers, not by underlying software

NoSQL Database: NoSQL Examples

- **Map Reduce Patent:**
  - Google granted US Patent 7,650,331, January 2010
    System and method for efficient large-scale data processing

    A large-scale data processing system and method includes one or more application-independent map modules configured to read input data and to apply at least one **application-specific map operation** to the input data to produce intermediate data values, wherein the map operation is automatically parallelized across multiple processors in the parallel processing environment. A plurality of intermediate data structures are used to store the intermediate data values. One or more application-independent reduce modules are configured to retrieve the intermediate data values and to apply at least one **application-specific reduce operation** to the intermediate data values to provide output data.

NoSQL Database: NoSQL Examples

- **Storing & Modifying Data:**
  - **Syntax varies**
    - HTML
    - Java Script
    - Etc.
  - **Asynchronous** – Inserts and updates do not wait for confirmation
  - Versioned
  - **Optimistic Concurrency**

NoSQL Database: NoSQL Examples

- **Retrieving Data:**
  - **Syntax Varies**
    - No set-based query language (i.e., it is bag semantics)
    - Procedural program languages such as Java, C, etc.
  - Application specifies retrieval path
  - No query optimizer
  - Quick answer is important
  - May not be a single “right” answer

NoSQL Database: NoSQL Examples

- **Open Source:**
  - Small upfront software costs
  - Suitable for large scale distribution on commodity hardware

NoSQL Database: NoSQL Examples

- **NoSQL Summary:**

  - **NoSQL databases reject:**
    - Overhead of ACID transactions
    - “Complexity” of SQL
    - Burden of up-front schema design
    - Declarative query expression
    - Yesterday’s technology
  - **Programmer responsible for:**
    - Step-by-step procedural language
    - Navigating access path

Summary

Summary

- **SQL Databases**
  - Predefined Schema
  - Standard definition and interface language
  - Tight consistency
  - Well defined semantics

- **NoSQL Database**
  - No predefined Schema
  - Per-product definition and interface language
  - Getting an answer quickly is more important than getting a correct answer

END

NoSQL Database: NoSQL Examples

- **CouchDB JSON Example:**

```
  "_id": "guid goes here",
  "_rev": "314159",

  "type": "abstract",

  "author": "Keith W. Hare"

  "title": "SQL Standard and NoSQL Databases",

  "body": "NoSQL databases (either no-SQL or Not Only SQL)
           are currently a hot topic in some parts of
           computing.",
  "creation_timestamp": "2011/05/10 13:30:00 +0004"
}
```

NoSQL Database: NoSQL Examples

- **CouchDB JSON Tags:**
  - "_id"
    - GUID – Global Unique Identifier
    - Passed in or generated by CouchDB
  - "_rev"
    - Revision number
    - Versioning mechanism
  - "type", "author", "title", etc.
    - Arbitrary tags
    - Schema-less
    - Could be validated after the fact by user-written routine