COSC 430
Advanced Database Topics

Lecture 1: Data Modeling: An Overview

Haibo Zhang
Computer Science, University of Otago
Today’s Focus

• Course introduction
  – Course goals
  – Teaching team
  – Course outline
  – Assessment

• Overview of the evolution of database data models
  – Hierarchical model
  – Network model
  – Relational model
  – Object model
  – Non-relational model
Course Goals

• Understand advanced database theory and research topics
  – advanced data models
  – distributed, timeseries, embedded, graph databases
  – data mining
  – new and emerging database technologies

• Perform practical exercises
  – database administration
  – Cassandra and Neo4j

• Develop research abilities
  – Paper reading, critical thinking, problem solving, report writing
Teaching Team

• **David Eyers** (course coordinator)
  – Room 1.25
  – Email: dme@cs.otago.ac.nz

• **Haibo Zhang**
  – Room 2.47
  – Email: haibo@cs.otago.ac.nz

• **Cathy Chandra**
  – Room 1.21 (member our IT team)
  – Email: cathy@cs.otago.ac.nz
Textbook

• No recommended textbooks

• Reading materials and online references will be provided in due course
Assessments

• Assignments (40%)
  – Assignment 1: relational theory (8%)
    • due @4pm on 22 March (Friday)
  – Assignment 2: Oracle database administration (15%)
    • Part 1 is due @4pm on 29 March (Friday)
    • Part 2 is due @4pm on 10 May (Friday)
  – Assignment 3: project (17%)
    • due @4pm on 27 May (Monday)

  Details on assignments are given in course webpage

• Examination (60%)
# Course Outline

<table>
<thead>
<tr>
<th>Week</th>
<th>Lecturer</th>
<th>Lecture topic</th>
<th>note</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Haibo</td>
<td>Data Modeling Overview</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Haibo</td>
<td>Relational Model</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Cathy</td>
<td>Oracle DBA Lab</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>David</td>
<td>NoSQL Models</td>
<td>Spanner paper</td>
</tr>
<tr>
<td>5</td>
<td>Haibo</td>
<td>Distributed Models</td>
<td>Spanner discussion</td>
</tr>
<tr>
<td>6</td>
<td>Haibo</td>
<td>Cassandra Lab</td>
<td>Apriori paper</td>
</tr>
<tr>
<td>7</td>
<td>Haibo</td>
<td>Data Mining</td>
<td>Apriori discussion</td>
</tr>
<tr>
<td>8</td>
<td>David</td>
<td>Time series database</td>
<td>Gorilla discussion</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>--- Easter Break ---</td>
</tr>
<tr>
<td>9</td>
<td>David</td>
<td>Graph database</td>
<td>Trinity discussion</td>
</tr>
<tr>
<td>10</td>
<td>David</td>
<td>Neo4j Lab</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>David</td>
<td>Embedded/ORMs Lab</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>David</td>
<td>Temporal and geographic databases</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>Guest Lecture/Question &amp; Answer/Revision</td>
<td></td>
</tr>
</tbody>
</table>
Academic Integrity and Misconduct

• Academic Integrity Policy
  – Being honest in your studying and assessments
  – www.otago.ac.nz/administration/policies/otago116838.html

• Student Academic Misconduct Procedures
  – Types of misconduct include plagiarism, copying, unauthorized collaboration, using unauthorised material, assisting someone else’s misconduct, etc.
  – http://www.otago.ac.nz/administration/policies/otago116850.html
Learning Objectives

• You should understand
  – What is a data model for database design and why is it needed?
  – What is data independence and why is it important?
  – Evolution of data models
    • Hierarchical model
    • Network model
    • Relational model
    • Object-relational model
    • NoSQL models
  – What is the Object-relational Impedance Mismatch?
  – What are the pros and cons of schema and schemaless designs?
Simplified Database System Environment

Users/Programmers

Database System

Application Programs/Queries

Database definition (Meta-data)

Software to Process Queries/Programs

Software to Access Stored Data

Stored Database
An Example of Relational Database

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>FNAME</td>
<td>MINIT</td>
<td>LNAME</td>
<td>SSN</td>
<td>BDATE</td>
<td>ADDRESS</td>
<td>SEX</td>
<td>SALARY</td>
<td>SUPERSSN</td>
<td>DNO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>John</td>
<td>B</td>
<td>Smith</td>
<td>123456789</td>
<td>9-Jan-1965</td>
<td>731 Fondren, Houston, TX</td>
<td>M</td>
<td>30000</td>
<td>333445555</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Franklin</td>
<td>T</td>
<td>Wong</td>
<td>333445555</td>
<td>8-Dec-1955</td>
<td>638 Voss, Houston, TX</td>
<td>M</td>
<td>40000</td>
<td>888665555</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alicia</td>
<td>J</td>
<td>Zelaya</td>
<td>999887777</td>
<td>19-Jul-1968</td>
<td>3321 Castle, Spring, TX</td>
<td>F</td>
<td>25000</td>
<td>987654321</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jennifer</td>
<td>S</td>
<td>Wallace</td>
<td>987654321</td>
<td>20-Jun-1941</td>
<td>291 Berry, Bellaire, TX</td>
<td>F</td>
<td>43000</td>
<td>888665555</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ramesh</td>
<td>K</td>
<td>Narayan</td>
<td>666884444</td>
<td>15-Sep-1962</td>
<td>975 Fire Oak, Humble, TX</td>
<td>M</td>
<td>38000</td>
<td>333445555</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Joyce</td>
<td>A</td>
<td>English</td>
<td>453453453</td>
<td>31-Jul-1972</td>
<td>5631 Rice, Houston, TX</td>
<td>F</td>
<td>25000</td>
<td>333445555</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ahmad</td>
<td>V</td>
<td>Jabbar</td>
<td>987987987</td>
<td>29-Mar-1969</td>
<td>980 Dallas, Houston, YX</td>
<td>M</td>
<td>25000</td>
<td>987654321</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>James</td>
<td>E</td>
<td>Borg</td>
<td>888665555</td>
<td>10-Nov-1937</td>
<td>450 Stone, Houston, TX</td>
<td>M</td>
<td>55000</td>
<td>NULL</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| DEPARTMENT |         |         |         |         |         |         |         |         |         |         |         |         |         |         |         |
| DNAME      | DNUMBER | MGRSSN  | MGRSTARTDATE |
| Research   | 5        | 333445555 | 22-May-1988 |
| Administration | 4    | 987654321 | 1-Jan-1995 |
| Headquarters | 1     | 888665555 | 19-Jun-1981 |
| Dummies    | 0        | 111100000 | 31-Dec-2004 |

<table>
<thead>
<tr>
<th>PROJECT</th>
<th>PNAME</th>
<th>PNUMBER</th>
<th>PLOCATION</th>
<th>DNUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProductX</td>
<td>1</td>
<td>Bellaire</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>ProductY</td>
<td>2</td>
<td>Sugarland</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>ProductZ</td>
<td>3</td>
<td>Houston</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Computerisation</td>
<td>10</td>
<td>Stafford</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Reorganisation</td>
<td>20</td>
<td>Houston</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>NewBenefits</td>
<td>30</td>
<td>Stafford</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DEPENDENT</th>
<th>ESSN</th>
<th>DEPENDENT_NAME</th>
<th>SEX</th>
<th>BDATE</th>
<th>RELATIONSHIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>333445555</td>
<td>Alice</td>
<td>F</td>
<td>5-Apr-1986</td>
<td>Daughter</td>
<td></td>
</tr>
<tr>
<td>333445555</td>
<td>Theodore</td>
<td>M</td>
<td>25-Oct-1983</td>
<td>Son</td>
<td></td>
</tr>
<tr>
<td>333445555</td>
<td>Joy</td>
<td>F</td>
<td>3-May-1958</td>
<td>Spouse</td>
<td></td>
</tr>
<tr>
<td>987654321</td>
<td>Abner</td>
<td>M</td>
<td>28-Feb-1942</td>
<td>Spouse</td>
<td></td>
</tr>
<tr>
<td>123456789</td>
<td>Michael</td>
<td>M</td>
<td>4-Jan-1988</td>
<td>Son</td>
<td></td>
</tr>
<tr>
<td>123456789</td>
<td>Alice</td>
<td>F</td>
<td>30-Dec-1988</td>
<td>Daughter</td>
<td></td>
</tr>
<tr>
<td>123456789</td>
<td>Elizabeth</td>
<td>F</td>
<td>5-May-1967</td>
<td>Spouse</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>WORKS_ON</th>
<th>ESSN</th>
<th>PNO</th>
<th>HOURS</th>
</tr>
</thead>
<tbody>
<tr>
<td>123456789</td>
<td>1</td>
<td>32.5</td>
<td></td>
</tr>
<tr>
<td>123456789</td>
<td>2</td>
<td>7.5</td>
<td></td>
</tr>
<tr>
<td>666884444</td>
<td>3</td>
<td>40.0</td>
<td></td>
</tr>
<tr>
<td>453453453</td>
<td>1</td>
<td>20.0</td>
<td></td>
</tr>
<tr>
<td>453453453</td>
<td>2</td>
<td>20.0</td>
<td></td>
</tr>
<tr>
<td>333445555</td>
<td>2</td>
<td>10.0</td>
<td></td>
</tr>
<tr>
<td>333445555</td>
<td>3</td>
<td>10.0</td>
<td></td>
</tr>
<tr>
<td>333445555</td>
<td>10</td>
<td>10.0</td>
<td></td>
</tr>
<tr>
<td>333445555</td>
<td>20</td>
<td>10.0</td>
<td></td>
</tr>
<tr>
<td>999887777</td>
<td>30</td>
<td>30.0</td>
<td></td>
</tr>
<tr>
<td>999887777</td>
<td>10</td>
<td>10.0</td>
<td></td>
</tr>
<tr>
<td>987987987</td>
<td>10</td>
<td>35.0</td>
<td></td>
</tr>
<tr>
<td>987987987</td>
<td>30</td>
<td>5.0</td>
<td></td>
</tr>
<tr>
<td>987654321</td>
<td>30</td>
<td>20.0</td>
<td></td>
</tr>
<tr>
<td>987654321</td>
<td>20</td>
<td>15.0</td>
<td></td>
</tr>
<tr>
<td>888665555</td>
<td>20</td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>
Data Model

• **A data model** is an abstract model that describes how data is organized/represented in an information system or a database management system.
  – is a system of concepts and their interrelations
  – is the “language” used to describe data
  – defines syntax and semantics
  – defines operations for data manipulation

• Why do we need data models?

• **database schema**: the description of a database
Database Data Models

- Tightly coupled with DBMS – implements one or more data models.

**Conceptual data model**
- concepts and relationships, ER model

**Implementation data model**
- logical model without storage details

**Physical data model**
- storage details

Database Design Process

1. Requirements Collection and Analysis
   - data requirements
2. Conceptual Design
   - Conceptual Schema (in a high-level data model)
3. Logical Design (Data Model Mapping)
   - Logical (Conceptual) Schema (in a high-level data model)
4. Physical Design
Figure 7.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 7.14.
Logical Schema: Example

Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.
Level of Data Abstraction

The ANSI/SPARC architecture

External schema

External schema

External schema

External/Conceptual Mapping

Conceptual Schema

Conceptual/Internal Mapping

Internal Schema

Disk

schema seen by apps

describes stored data in terms of data model

storage details file organization indexes

COSC430
Lecture 1: Course introduction + Evolution of Data models
Data Independence

• **Logical data independence**
  – The capacity to change the conceptual schema without having to change external schemas

• **Physical data independence**
  – The capacity to change the internal schema without having to change the conceptual schemas

• **Why are these properties important?**
  Reduce maintenance cost since
  • logical design can change over time
  • physical design needs to be tuned for better performance
Evolution of Data Models

Hierarchical data model

- IBM’s database program IMS – Information Management System
  - **Record type**: a collection of named fields
  - Each record type must have a key
  - Record types must be arranged in a **tree structure**

- **Tree-structure has two drawbacks**
  - Redundant data
  - Existence depends on parents

---

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1960</td>
<td>Supplier (sno, pname, qty)</td>
</tr>
<tr>
<td>1970</td>
<td>Part (pno, pname, psize)</td>
</tr>
<tr>
<td>1980</td>
<td>Purchase(pcno, qty)</td>
</tr>
</tbody>
</table>
Evolution of Data Models

Hierarchical data model

- **Data manipulation language: DL/1**
  - Each record has a hierarchical sequence key (HSK)
  - Records are ordered: depth-first and left-to-right
  - HSK defines semantics of commands
    - `get_next`
    - `get_next_within_parent`
  - A **record-at-a-time language**
    - Programmer constructs for solving and optimizing query

- **Data independence**
  - Very limited physical independence
    - Records are stored sequentially based on HSK
  - Some logical independence
    - DL/1 programs run on logical database
Evolution of Data Models

Network Data Model

• **Also known as CODASYL (COnference on DAta SYstems Languages) data model**

• **Similar to hierarchical model**
  – Collection of record types, each with keys
  – Record-at-a-time data manipulation language

• **Different from hierarchical model**
  – Record types are organized into a network
    • A given record instance can have multiple parents
    • At least one entry point to the network graph
Evolution of Data Models

Network Data Model

- It provides more flexibility but is more complex than the hierarchical model
  - Navigate a hierarchical space in the hierarchical model
  - Navigate a multi-dimensional space in the network model

- Data independence
  - No physical independence
  - No logical independence
Evolution of Data Models

Relational Data Model

- Invented by E.F. Codd, IBM Research, in 1970
  - A simple data structure (relation, table)
  - A high-level **set-at-a-time** data manipulation language
  - Define logical schema only, no physical schema

- Entity-Relationship (E-R) model (Peter Chen, 1970)
  - Entities
  - Attributes
  - Relationships

![Diagram](image_url)
Evolution of Data Models

Relational Data Model

• Has the flexibility to represent almost anything

• **Set-at-a-time** DML offers substantial advantages over record-at-a-time DML

• **Data independence**
  – **Physical** independence
    Yes, no specification of what storage looks like.
  – **Logical** independence
    Yes, through views
Object-relational Impedance Mismatch

• Problems encountered when relational model and object model work together
  – Difference between the relational model and the in-memory data structures
  – Object-relational mapping (e.g. Hibernate and iBATIS)
Evolution of Data Models

Object-oriented data model

• Reasons for OO model
  – Need for more complex applications
  – Need for additional data modelling features
  – Popularity of OO programming languages

• Object
  – Components: state (value) and behaviour (operations)
  – It can have a complex structure as well as specific operations defined by the programmer.

• Has some commercial products (e.g. O2, Objectstore), but did not make much impact on mainstream data management
Evolution of Data Models

Object-oriented data model

• Example of class definition

```plaintext
define class DEPARTMENT

  type tuple (  
    Dname: string;
    Dnumber: integer;
    Mgr: tuple (  
      Manager: EMPLOYEE;
      Start_date: DATE;
    );
    Locations: set (string);
    Employees: set (EMPLOYEE);
    Projects: set (PROJECT);
  );

  operations
  no_of_emps: integer;
  create_dept: DEPARTMENT;
  destroy_dept: boolean;
  assign_emp(e: EMPLOYEE): boolean;
  (* adds an employee to the department *)
  remove_emp(e: EMPLOYEE): boolean;
  (* removes an employee from the department *)

end DEPARTMENT;
```
### Evolution of Data Models

- **Object-relational data model**
  - Relation is still the fundamental structure
    - Allow of nested relations

#### Example of N1NF

<table>
<thead>
<tr>
<th>ID</th>
<th>person</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>fname</td>
<td>mname</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evolution of Data Models

Object-relational data model

• Relation is still the fundamental structure
  – Allow of nested relations

• Extend the relational model to the OO domain
  – Type system with user-defined types (UDT)
    • Including set, bag, array, list collection types
    • Including structures like records
    • Using type inheritance
  – Methods
    • Special operations can be defined over the UDTs
    • Special operators for complex types, e.g. images
  – References
    • Several ways for references and de-references, e.g. using pointers to avoid redundancy

• Implemented in most popular relational DBMS
Evolution of Data Models

Object-relational data model

- Example of creating an object type in Oracle

```sql
CREATE TYPE person_typ AS OBJECT (  
idno NUMBER,  
first_name VARCHAR2(20),  
last_name VARCHAR2(25),  
email VARCHAR2(25),  
phone VARCHAR2(20),  
MAP MEMBER FUNCTION get_idno RETURN NUMBER,  
MEMBER PROCEDURE display_details ( SELF IN OUT NOCOPY person_typ ) );
/

CREATE TYPE BODY person_typ AS  
MAP MEMBER FUNCTION get_idno RETURN NUMBER IS  
BEGIN  
  RETURN idno;  
END;  
MEMBER PROCEDURE display_details ( SELF IN OUT NOCOPY person_typ ) IS  
BEGIN  
  -- use the PUT_LINE procedure of the DBMS_OUTPUT package to display details  
  DBMS_OUTPUT.PUT_LINE(TO_CHAR(idno) || ' ' || first_name || ' ' || last_name);  
  DBMS_OUTPUT.PUT_LINE(email || ' ' || phone);  
END;  
END;  
/
```
Different Types of Data

• **Structured Data**
  – Has a strict format, highly organized, conforms to a schema
  – Example: data stored in SQL database
  – Represents only 5-10% of data

• **Semi-structured Data**
  – May have certain structure but not all information collected has identical structure
  – Example: XML and JSON documents
  – Represents another 5-10% of data

• **Unstructured Data**
  – Has no structure
  – Example: text, video, image
  – Represents around 80% of data
The Big Data Era

• What is big data?

  – Big Data is high-volume, high-velocity, and/or high-variety information asset that requires new forms of processing to enable enhanced decision making, insight discovery and process optimization. (Gartner 2012)

  – Big data is data set that are so voluminous and complex that traditional data processing application software are inadequate to deal with them. (Wikipedia)
Big Data Characteristics

Data produced by Facebook everyday:
500+ terabytes of data each day
2.7 billion Like actions
300 million photos
Scans roughly 105 terabytes of data each half hour.
RDBMS for Big Data?

- More data than we are used to handling
  - Not always extremely high volumes
  - Not sure what to analyze!
- Data is unstructured
  - Data does not fit into conventional storage
- Poor scalability

![Diagram showing cost/performance trade-off over time, indicating that today's doable and expensive databases are out of reach for tomorrow's vast volumes.](image-url)
Attack of Clusters

• A shift from **scale-up to scale-out**
  – Computer architectures based on cluster of commodity hardware emerged as the only solution to deal with the explosion of data volume.
  – But relational databases are not designed to run (and do not work well) on clusters!

• The mismatch between relational database and clusters motivates some organization to develop alternative solutions to data storage
  – BigTable (Goolge)
  – Dynamo (Amazon)
Evolution of Data Models

**NoSQL**

- The term appeared in the late 90s (Carlo Strozzi)
  - Open-source relational database
  - Tables with ASCII files without SQL, but still relational
- **Current interpretation**
  - Open-source, distributed, *nonrelational* database
  - Johan Oskarsson, 2009
  - Main features:
    - Not using SQL and the relational model
    - Open-source
    - Running on clusters
    - Schemaless
  - No accepted precise definition
  - Most people say that NoSQL means "Not Only SQL"
Evolution of Data Models

NoSQL

• Four Main Data models

  – Key-value

  – Document

  – Column-family

  – Graph
Evolution of Data Models

NoSQL

- **Key-value**
  - a collection <key,value> pairs
  - A hash table or hash map
  - Values can be complex compound structure and objects
  - Look up based on the key: simple with no joins and foreign key constraints
  - Easy to distribute across a cluster
Evolution of Data Models

NoSQL

- **Document**
  - a collection <key, document > pairs
  - a document is a block of data with some format or encoding such as JSON and XML
  - can have the same or different structure
  - no schemas
  - allow to query based on content or metadata

```json
{
  "name":"John",
  "age":30,
  "cars": {
    "car1":"Ford",
    "car2":"BMW",
    "car3":"Fiat"
  }
}
```

```xml
<shipto>
  <name>Ola Nordmann</name>
  <address>Langgt 23</address>
  <city>4000 Stavanger</city>
  <country>Norway</country>
</shipto>
```
Evolution of Data Models

NoSQL

• **Column-family**
  – a column-family contains multiple rows
  – each row contains a number of columns
  – each column contains a name, a value and timestamp

```plaintext
Row Key
Name
Value
Timestamp

Column
Name
Value
Timestamp

Bob
emailAddress
bob@example.com
1465676582

gender
male
1465676582

age
35
1465676582
```
Evolution of Data Models

NoSQL

- Column-family
  - a keyspace is analogous to a schema that contains all column families for one application.
Evolution of Data Models

NoSQL

- **Column-family benefits**
  - Efficient at data compression and partitioning
  - Perform well with aggregation queries such as SUM, AVG, etc.
  - Has the scalability to spread data across a large cluster of machines
  - Fast to load and query data
Relationship Modeling

• Relational data model
  – Relationships exit only between tables (foreign keys), not good for highly connected domains
  – Special checking for nullable columns
  – Relationship traversal can be very expensive (joins)

• NoSQL data model (key-value, document, column)
  – Store set of disconnected values/documents
  – Embed an aggregate’s identifier inside the field belonging to another aggregate (join at application level, expensive)
Relationship Modeling

NOSQL Databases Also Lack Relationships

Most NOSQL databases—whether key-value-, document-, or column-oriented—store sets of disconnected documents/values/columns. This makes it difficult to use them for connected data and graphs.

One well-known strategy for adding relationships to such stores is to embed an aggregate's identifier inside the field belonging to another aggregate—effectively introducing foreign keys. But this requires joining aggregates at the application level, which quickly becomes prohibitively expensive.

When we look at an aggregate store model, such as the one in Figure 2-3, we imagine we can see relationships. Seeing a reference to order: 1234 in the record beginning user: Alice, we infer a connection between user: Alice and order: 1234. This gives us false hope that we can use keys and values to manage graphs.

Figure 2-3. Reifying relationships in an aggregate store

In Figure 2-3 we infer that some property values are really references to foreign aggregates elsewhere in the database. But turning these inferences into a navigable structure doesn't come for free, because relationships between aggregates aren't first-class citizens in the data model—most aggregate stores furnish only the insides of NOSQL Databases Also Lack Relationships | 15

Example from O'Reily Graph Databases
Evolution of Data Models

NoSQL

• **Graph**
  – Data is stored as **connected data** using a graph structure
  – The Labeled Property Graph Model
    • **Nodes**: contain properties, with one or more labels
    • **Relationships**: connect nodes and structure the graph, directed, can have properties
    • **Properties**: key-value pairs
    • **Labels**: group nodes together

Using this graph, we can determine the various paths of a forwarded email chain.

MATCH (email:Email {id: '11'})<-[:FORWARD_OF*]-(:Forward)
RETURN count(f)
Evolution of Data Models

• **Graph: Advantages**
  
  – Explicit graph structure: semantic dependencies are made explicit
  
  – New nodes and new relationships can be easily added without data migration and restructuring.
  
  – Relationships correspond to paths; querying the database = traversing the graph
  
  – Schema-free
  
  – Suitable to model complex, highly-connected data (e.g. social networks, web data, product preference)
Schemaless Databases

• Schemaless – a common theme across all forms of NoSQL databases.
  
  – Key-value stores allow to store any data under a key.
  
  – Document databases make no restriction on the structure of the documents.
  
  – Column-family databases allow to store any data under any column.
  
  – Graph databases allow to freely add new nodes, properties and relationships.
Schema vs Schemaless

• Schema
  – Global data definition (data types, constraints)
  – Optimal access (store, manage, and access)
  – Less flexibility (maintain schema, be aware of data types)

• Schemaless
  – Flexibility (any kind of data, easily change data organization)
  – Ease of use and maintenance
  – Poor integrity
  – Performance suffers (implicit schema shifted to the application code)
Summary

• Concept of data model
• Data abstraction and independence
• Evolution of data models
  – Hierarchical data model
  – Network data model
  – Relational data model
  – Key-value store
  – Document
  – Column-family
  – Graph
• Big data characteristics
• Schemaless database
Reading Materials

• Carlos Coronel, Steven Morris and Peter Rob
  Database Systems Design,
  Implementation, and Management (13e)
  Chapter 2

• Pramod J. Sadalage and Martin Fowler
  NoSQL Distilled: A Brief Guide to the
  Emerging World of Ployglot Persistence
  Chapters 1-3