Representing and Storing Structured Data

LBSC 690: Jordan Boyd-Graber

October 15, 2012

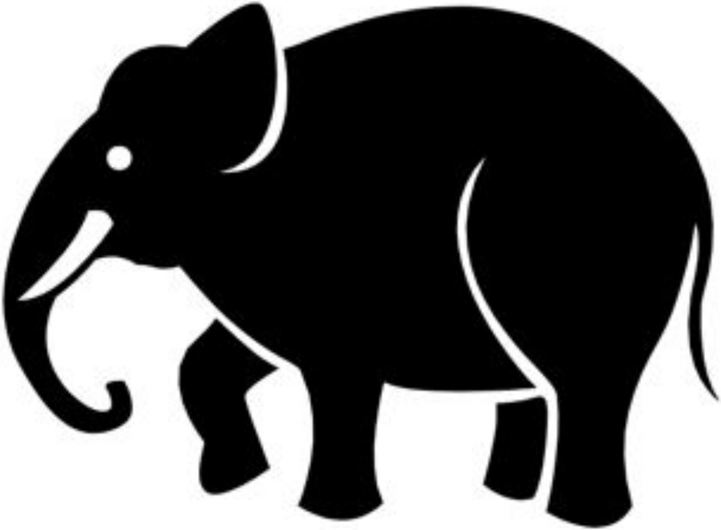
COLLEGE OF INFORMATION STUDIES

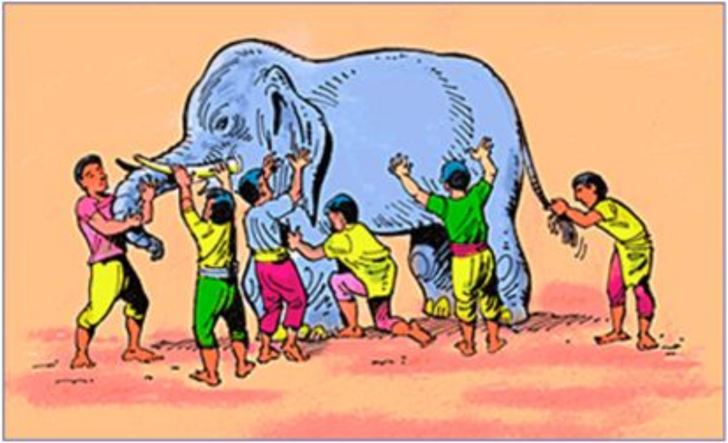
Adapted from Jimmy Lin's Slides

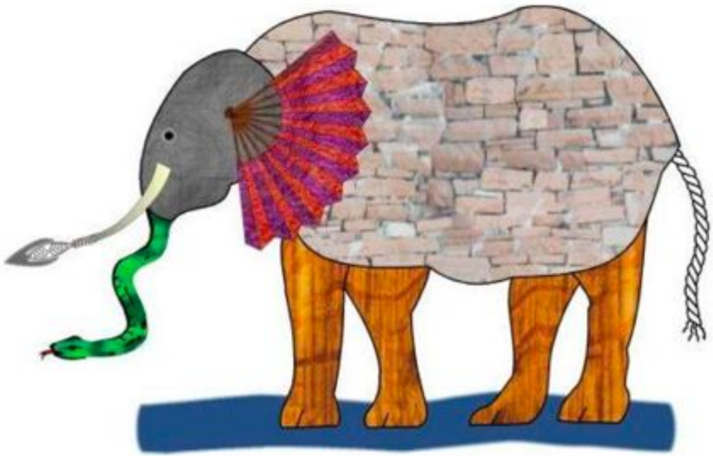
Goals

- Metadata: XML
- Databases

Outline







Take-Away Messages

- Metadata makes data useful
- XML is a way to encode data and metadata
- XML allows computers to exchange information in new and interesting ways

7/1/1988	OL	950	20.3	13	0.8	-0.1	33.1	27.8	5.3	5.92
7/2/1988	OL	950	24.2	12.6	1	-0.1	27.8	23.9	3.8	4.56
7/3/1988	OL	.	.	.	.	.	.	.	.	.
7/4/1988	OL	950	0.4	16.3	0.4	0.2	41	34.5	6.5	15.5
7/5/1988	OL	1005	32.9	18.9	1.4	0.3	29.8	23.7	6.1	14.23
7/6/1988	OL	1020	32.3	20.5	1.4	0.3	23.4	18.9	4.5	12.97
7/7/1988	OL	1015	36.8	24.9	1.7	0.5	18.6	15.3	3.2	13.92
7/8/1988	OL	925	42.8	25.6	2.5	0.6	23.7	19.9	3.9	15.18
7/9/1988	OL	945	23.3	27.8	0.7	0.8	27.7	23.5	4.3	12.33
7/10/1988	OL	1030	49.8	26.2	2.6	0.6	40.3	34	6.3	22.14
7/11/1988	OL	940	44.8	25.2	2.5	0.8	34	29.2	4.8	16.76
7/12/1988	OL	1010	47.6	26.9	2.6	0.7	47.3	39.6	7.7	16.13
7/13/1988	OL	945	36.5	22.6	1.9	0.6	36.7	32.6	4	15.5
7/14/1988	OL	950	19.5	18.6	0.4	0.5	302	39.1	262.9	11.07
7/15/1988	OL	955	31.7	15.7	1.5	0.4	29.7	25	4.7	9.49
7/16/1988	OL	955	23.3	14.5	1.8	0.8	23.4	20.7	2.7	8.14
7/17/1988	OL	1015	23.8	16.6	1.6	0.6	27.7	24.1	3.7	9.17
7/18/1988	OL	934	32.9	16.7	2.1	0.7	34	28.9	5.1	9.49
7/19/1988	OL	1010	29.2	20.4	1.9	0.7	26	22.3	3.7	10.44
7/20/1988	OL	952	44.8	24.8	2.1	0.8	31.7	27.5	4.2	10.75
7/21/1988	OL	1029	33.7	37.1	1.9	0.6	34.5	30.1	4.3	12.02
7/22/1988	OL	1017	34.3	32.9	2	0.7	31.4	26.2	5.1	12.65
7/23/1988	OL	1040	35.7	24.6	2	0.8	23.7	20.4	3.3	15.5
7/24/1988	OL	923	47.6	28.9	2.9	0.8	67.3	58.9	8.4	20.87
7/25/1988	OL	1030	58.3	32.6	2.9	0.7	68	59.3	8.7	22.14
7/26/1988	OL	950	49.3	29.2	3.4	0.6	86	75.1	10.9	21.19
7/27/1988	OL	1006	54.1	20.9	3.9	0.6	94	82.8	11.2	25.06
7/28/1988	OL	1010	40.5	16.5	1.7	0.3	41	34.4	6.6	6.54
7/29/1988	OL	1000	25.5	23.6	1.4	0.1	41	35.4	5.6	3.82
7/30/1988	OL	1005	47.9	17.6	0.8	0.1	18.3	15.9	2.3	4.19
7/31/1988	OL	1015	38	22.5	1.5	0.1	30	25.3	4.7	4.44
8/1/1988	OL	1018	21.2	8.8	1.1	-0.1	24.7	21.1	3.6	4.81
8/2/1988	OL	1004	38.5	22.8	2.1	0.3	54	46.8	7.2	9.8
8/3/1988	OL	1011	94	32.6	2.1	0.3	45.5	38.9	6.6	9.49
8/4/1988	OL	955	58.3	43.1	2.5	1.1	41	33.1	7.9	9.8
8/5/1988	OL	951	55.8	42.2	2.1	0.8	38	31	7	8.86

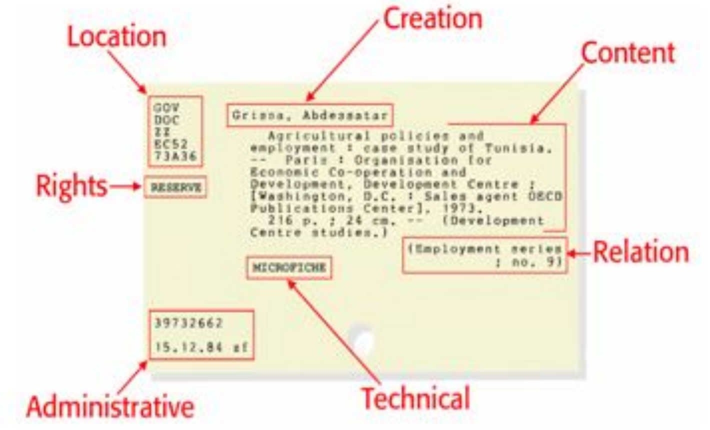
What's going on here? How do I use this?

Metadata

Literally "data about data"

"a set of data that describes and gives information about other data" – Oxford English Dictionary

Dublin Core



Location
Creation
Content
Rights
Relation
Administrative
Technical
GOV
DOC
ZZ
EC52
73A36
RESERVE
Grissa, Abdessatar
Agricultural policies and employment : case study of Tunisia. -- Paris : Organisation for Economic Co-operation and Development, Development Centre ; [Washington, D.C. : Sales agent OECD Publications Center], 1973.
216 p. ; 24 cm. -- (Development Centre studies.)
(Employment series ; no. 9)
MICROFICHE
39732662
15.12.84 zf

What is the Dublin Core?

- A metadata standard for describing digital resources
- An initiative to create a library card catalog for the Web
- Dublin Core fields:

Title	Creator	Subject
Description	Publisher	Contributor
Date	Type	Format
Identifier	Source	Language
Relation	Coverage	Rights

Encoding Metadata

- Language for encoding metadata should be:
  - Universal - so all can understand
  - Flexible - to incorporate different types
  - Extensible - flexible to custom types
  - Simple - to encourage adoption
  - Modular - so that schemes can be mixed, extended

How do we encode data for interoperability?

Challenges

January 31, 2001
31 janvier 2001
2001-01-31
01-31-2000
980942400

Outline

What is XML?

- XML = eXtensible Markup Language
- XML is a standard for exchanging structured data
  - Provides standardization at the syntactic level
  - Does not provide “meaning” for the tags
  - XML is a standard recommended by the W3C

Goals of XML

- Easy to use
- Easy to extend and adapt
- Easy to write programs that use XML
- Support a wide variety of applications
- Should be human legible
- Formal and concise

Refresher: Elements and Attributes

Attribute

```
<person age="28" />
```

Element

```
<person>
<age>28</age>
</person>
```

The Basic Rules

- XML is case sensitive
- All start tags must have end tags
- Elements must be properly nested
- XML declaration is the first statement

```
<?xml version="1.0"?>
```

- Every document must contain a root element
- Attribute values must have quotation marks

```
<item id="33905">
```

- Certain characters are reserved for parsing

```
\&lt; = ``<''
```

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.1/">

    <rdf:Description
       rdf:about="http://media.example.com/audio/guide.ra">

<dc:creator>Rose Bush</dc:creator>
<dc:title>A Guide to Growing Roses</dc:title>
<dc:description>Describes process for planting and nurturing
  different kinds of rose bushes.</dc:description>
<dc:date>2001-01-20</dc:date>
    </rdf:Description>

</rdf:RDF>
```

- What does XML do?

- What does XML do? . . . **nothing**
- Syntax vs. semantics
- XML vs HTML

Historic Perspective: Three Core Technologies

- HTTP - HyperText Transfer Protocol
  - A protocol for transferring data between machines on the Internet
- URL - Uniform Resource Locator
  - A scheme for referencing the specific location of a resource
- HTML - HyperText Markup Language
  - A markup language for encoding information to be read by humans

HTTP and URLs have stood the test of time.

But by 1996, HTML was already showing signs of age . . .

HTML

- Started with very few tags
- Language evolved as more tags were added:
  - Forms
  - Tables
  - Fonts
  - Frames
  - . . .

Problems with HTML

- I want personalized tags
  - HTML can't be extended
- I want to incorporate other types of data
  - Mathematics, database entries, literary text, poems, purchase orders ...
  - HTML can't accommodate other types of data
- I want to process pages automatically with software
  - HTML is too messy and inconsistent
  - Browsers are too forgiving

Back to Basics

- HTML was defined using SGML
  - Standard Generalized Markup Language
  - A meta-language for defining languages
  - Complex, sophisticated, powerful . . .

Back to Basics

- HTML was defined using SGML
  - Standard Generalized Markup Language
  - A meta-language for defining languages
  - Complex, sophisticated, powerful . . . too difficult to use
  - Idea: create a simpler version of SGML . . .

Back to Basics

- HTML was defined using SGML
  - Standard Generalized Markup Language
  - A meta-language for defining languages
  - Complex, sophisticated, powerful . . . too difficult to use
  - Idea: create a simpler version of SGML . . . the birth of XML!

XML Languages

- XML can be used to define other languages
- Many XML languages, optimized for different roles
  - XHTML: HTML by XML rules
  - MathML: for mathematics
  - EPUB: for creating eBooks
  - RSS: for news feeds
  - Civ IV: Create your own game
  - SVG: Create graphics

XHTML: Cleaning up HTML

```
<?xml version="1.0" encoding="iso-8859-1"?>
<html xmlns="http://www.w3.org/TR/xhtml1" >
<head>
   <title> Title of text XHTML Document </title>
</head>
<body>
<div class="myDiv">
    <h1> Heading of Page </h1>
     <p>A paragraph this one  with an
<img src="image.gif"  alt="waste␣of␣time" />
  image, and a <br /> line break. </p>
</div>
</body></html>
```

What's new?

New preamble to tell us what's here, and tags must have explicit ends.

MathML

An XML language for defining mathematic formulas

$$x^2 + 4x + 4 = 0$$

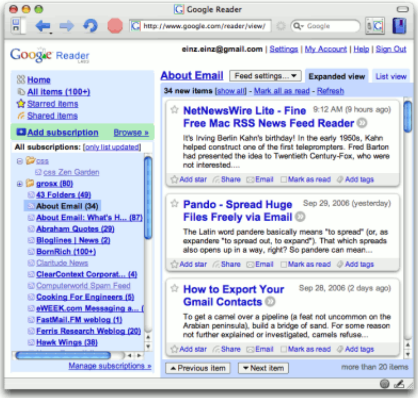
```
<mrow>
  <mrow>
<msup><mi>x</mi><mn>2</mn></msup>
<mo>+</mo>
<mrow>
  <mn>4</mn>
  <mo>&InvisibleTimes;</mo>
  <mi>x</mi>
</mrow>
    <mo>+</mo><mn>4</mn>
  </mrow>
  <mo>=</mo><mn>0</mn>
</mrow>
```

EPUB

- Format for putting books on mobile readers (except Kindles)
- Divide up a book into XHTML files
- Create two additional XML files
  - opf (open packaging format)
    - Metadata (using Dublin Core)
    - All the files needed
    - Linear reading order
  - ncx (navigation control file for XML)
    - Hierarchical organization of content (for easy navigation)

RSS

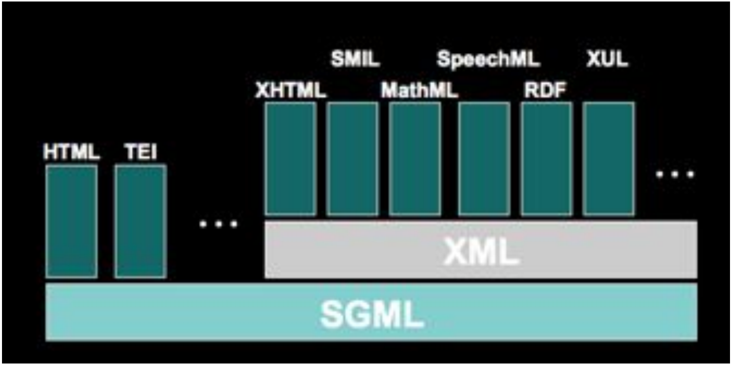
- RSS = Really Simple Syndication or Rich Site Summary
- An XML format for distributing news headlines on the Web



And Others ...

- CML: chemical Markup Lang
- CellML: biological models
- BSML: bioinformatic sequences
- MAGE-ML: Microarray Gene Expression
- XSTAR: for archaeological research
- MARCXML: MARC in XML
- AML: astronomy markup language
- SportsML: for sharing sports data
- List goes on and on and on ...

The XML Family Tree



Mixing XML Dialects

- XML is designed to support the integration of multiple standards
- Allows users to mix elements from different standards
  - Snapping together XML dialects like Lego pieces
  - Based on the notion of "namespaces"

Example

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rss="http://purl.org/rss/1.0/"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
 <rss:channel rdf:about="http://www.xml.com/xml/news.rss">
   <rss:title>XML.com</rss:title>
   <rss:link>http://xml.com/pub</rss:link>
   <dc:description>
     XML.com features a rich mix of
     information and services for the XML community.
   </dc:description>
   <dc:subject>XML, RDF, metadata, information
     syndication services</dc:subject>
   <dc:identifier>http://www.xml.com</dc:identifier>
   <dc:publisher>O'Reilly␣&␣Associates,␣Inc.</dc:publisher>
␣␣␣␣<dc:rights>Copyright␣2000,␣O'Reilly &
     Associates, Inc.</dc:rights>
 </rss:channel>
</rdf:RDF>
```

Another Example

```
<?xml version="1.0" encoding="iso-8859-1"?>
<html xmlns="http://www.w3.org/TR/xhtml1" >
<head>
   <title> Title of XHTML Document </title>
</head><body>
<div class="myDiv">
    <h1> Heading of Page </h1>
     <math xmlns="http://www.w3.org/1998/Math/MathML">
   ... MathML markup ...
     </math>
     <p> more html stuff goes here </p>
    <smil xmlns="http://www.w3.org/TR/smil1">
   ... SMIL markup ...
     </smil>
</div>
</body></html>
```

Outline

Interoperability

- What does it mean and what's the role of XML?

Interoperability

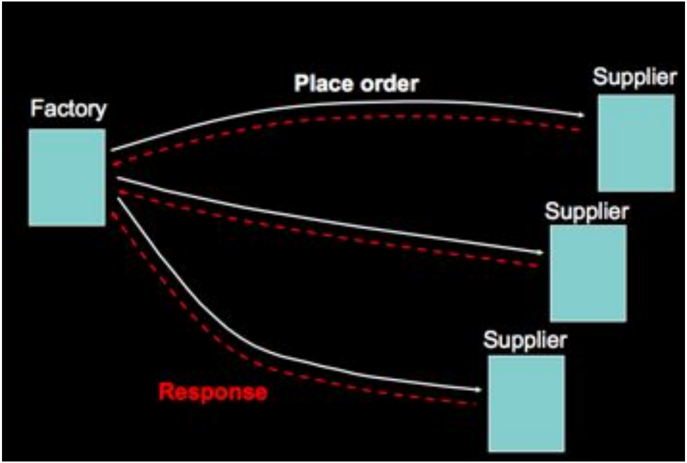
- What does it mean and what's the role of XML?

XML: universal format for data interchange

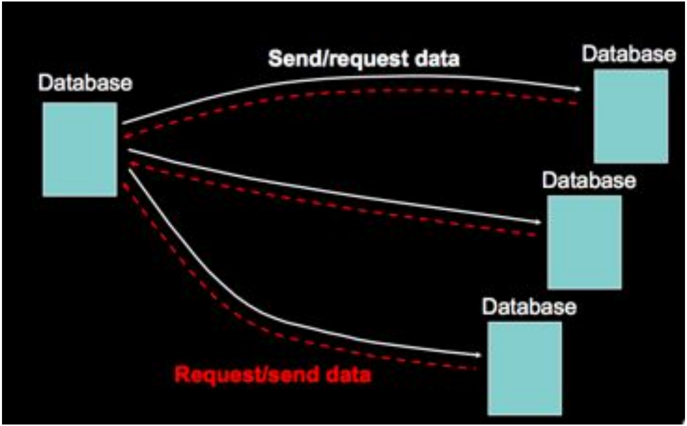
Software exchanges data as XML-format messages

- Advantages?
  - Eliminates proprietary data formats
  - Promotes interoperability
  - Encourages cooperation
  - Leverages lots of existing XML processing software

XML Messaging



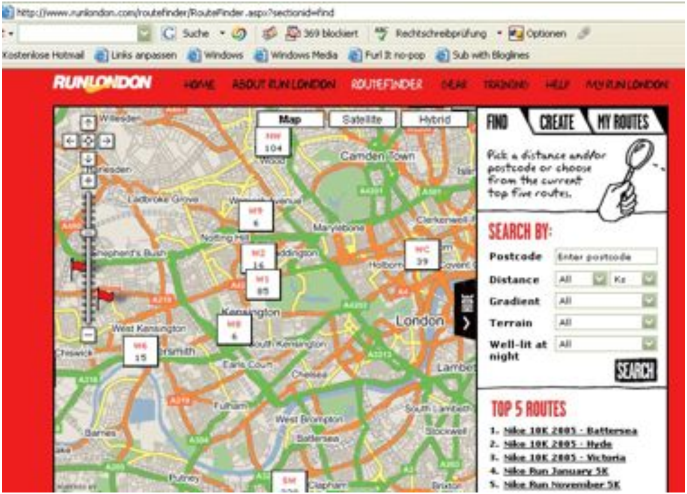
XML Messaging



What's in it for me?

- Webapps
  - Lower overhead
  - Richer data
  - More portability
- Mashups
- Syntax vs. Semantics

Mashups



RUNLONDON
HOME
ABOUT RUN LONDON
ROUTEFINDER
HELP
MY RUN LONDON
Map
Satellite
Hybrid
FIND
CREATE
MY ROUTES
Pick a distance and/or postcode or choose from the current top five routes.
SEARCH BY:
Postcode
Enter postcode
Distance
Gradient
Terrain
Well-lit at night
SEARCH
TOP 5 ROUTES
1. Nike 10K 2005 - Battersea
2. Nike 10K 2005 - Hyde
3. Nike 10K 2005 - Victoria
4. Nike Run January 5K
5. Nike Run November 5K
Camden Town
Ladbroke Grove
Notting Hill
Marylebone
Kensington
West Kensington
South Kensington
Chelsea
Fulham
West Brompton
Battersea
London
Putney
Clapham
Brixton

Mashups


Map
Satellite
Hybrid
EPISODE 55
A DECENT PROPOSAL
VIDEO
INFORM
SUSPECTS
LOCATION
GO TO NEXT LOCATION
March 12
LOCATION TOUR

XML Schema

- Defines what a valid XML document should look like
  - Fields
  - Attributes
  - Number of entries
- Has filename extension "xsd"
- There are plenty of XML validators out there
- Won't go into details ... think of it like a rulebook

Extensible Stylesheet Language Transformations

- XSLT transforms one XML document into another
- Often used to display XML to a user
  - Webpage
  - Graphics
- Syntax varies, semantics are fixed

Business Card: Source Data

```
<?xml version="1.0"?>

<card type="simple">
  <name>John Doe</name>
  <title>CEO, Widget Inc.</title>
  <email>john.doe@widget.com</email>
  <phone>(202) 456-1414</phone>
</card>
```

Business Card: XSLT Transformation

```
<xsl:stylesheet
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.
    xmlns="http://www.w3.org/1999/xhtml">

  <xsl:template match="card">
    <html>
      <head><title>business card</title></head>
      <body>
 <xsl:apply-templates select="name"/>
 <xsl:apply-templates select="title"/>
 <xsl:apply-templates select="email"/>
 <xsl:apply-templates select="phone"/>
      </body>
    </html>
  </xsl:template>
```

Business Card: XSLT Transformation (cont.)

```
  <xsl:template match="name">
    <h1><xsl:value-of select="text()"/></h1>
  </xsl:template>

  <xsl:template match="title">
    <b>Title:</b> <xsl:value-of select="text()"/> <br/>
  </xsl:template>

  <xsl:template match="email">
    <b>Email:</b> <a href="mailto:{text()}"><tt>
      <xsl:value-of select="text()"/>
    </tt></a><br/>
  </xsl:template>

  <xsl:template match="phone">
    <b>Phone:</b> <xsl:value-of select="text()"/> <br/>
  </xsl:template>

</xsl:stylesheet>
```

Card with Style

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="card_style1.xml"?>

<card type="simple">
  <name>John Doe</name>
  <title>CEO, Widget Inc.</title>
  <email>john.doe@widget.com</email>
  <phone>(202) 456-1414</phone>
</card>
```

In a browser . . .

John Doe

Title:CEO, Widget Inc.
Email:john.doe@widget.com
Phone:(202) 456-1414

XML isn't all there is

- S-Expressions
  - Based on logical statements
  - Not used outside academia (not in it too much, either)
- Protocol Buffers
  - Blazingly fast
  - More constrained than XML (have to specify data types, ranges)
- JSON
  - Designed specifically for web applications
  - Lighter weight than XML

Take-Away Messages

- Metadata makes data useful
- XML is a way to encode data and metadata
- XML allows computers to exchange information in new and interesting ways

Outline

Take-Away Messages

- Databases are suitable for storing structured information
- Databases are important tools to organize, manipulate, and access structured information
- Databases are integral components of modern Web applications

Definitions

Structured Information

What you put in a database (e.g. from XML)

Database

What you put structured information in.

Database Management System (DBMS)

Software system designed to store, manage, and facilitate access to databases

What's a database?

An integrated collection of data organized according to some model . . .

What's a relational database?

An integrated collection of data organized according to a relational model
...

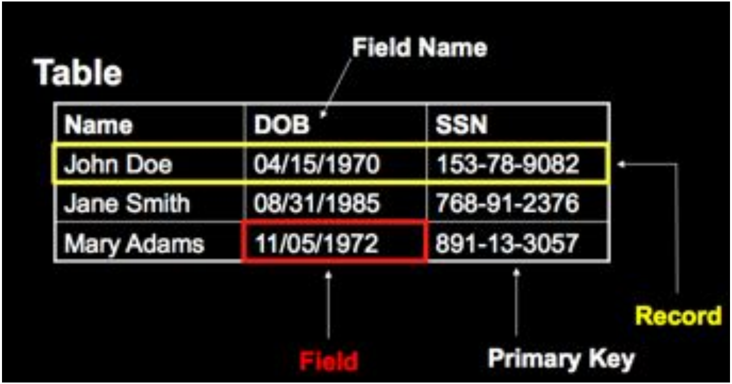
Databases (try to) model reality...

- **Entities**: things in the world (Example: airlines, tickets, passengers)
- **Relationships**: how different things are related (Example: the tickets each passenger bought)
- **"Business Logic"**: rules about the world (Example: fare rules)

Components of a Relational Database

- **Field**: an "atomic" unit of data
- **Record**: a collection of related fields
- **Table**: a collection of related records
  - Each record is a row in the table
  - Each field is a column in the table
- **Database**: a collection of tables

A Simple Example



Components of a Relational Database

Why "Relational?"

View of the world in terms of entities and relations between them

- Tables represent "relations"
- Each row in the table is sometimes called a "tuple"
- Each tuple is "about" an entity
- Fields can be interpreted as "attributes" or "properties" of the entity

Data is manipulated by "relational algebra":

- Defines things you can do with tuples
- Expressed in SQL (Structured Query Language, next week)

The Registrar Example

- What do we need to know?
  - Something about the students ?(e.g., first name, last name, email, department)
  - Something about the courses ?(e.g., course ID, description, enrolled students, grades)
  - Which students are in which courses
- How do we capture these things?

A first stab . . .

Put everything in a big table...

Student ID	Last Name	First Name	Dept ID	Dept	Course ID	Course name	Grade	email
1	Arrows	John	EE	EE	lbsc690	Information Technology	90	jarrows@wam
1	Arrows	John	EE	Elec Engin	ee750	Communication	95	ja_2002@yahoo
2	Peters	Kathy	HIST	HIST	lbsc690	Informatino Technology	95	kpeters2@wam
2	Peters	Kathy	HIST	history	hist405	American History	80	kpeters2@wma
3	Smith	Chris	HIST	history	hist405	American History	90	smith2002@glue
4	Smith	John	CLIS	Info Sci	lbsc690	Information Technology	98	js03@wam

A first stab . . .

Put everything in a big table...

Student ID	Last Name	First Name	Dept ID	Dept	Course ID	Course name	Grade	email
1	Arrows	John	EE	EE	lbsc690	Information Technology	90	jarrows@wam
1	Arrows	John	EE	Elec Engin	ee750	Communication	95	ja_2002@yahoo
2	Peters	Kathy	HIST	HIST	lbsc690	Informatino Technology	95	kpeters2@wam
2	Peters	Kathy	HIST	history	hist405	American History	80	kpeters2@wma
3	Smith	Chris	HIST	history	hist405	American History	90	smith2002@glue
4	Smith	John	CLIS	Info Sci	lbsc690	Information Technology	98	js03@wam

What's wrong with this?

Goals of "Normalization"

- Save space
  - Save each fact only once
- More rapid updates
  - Every fact only needs to be updated once
- More rapid search
  - Finding something once is good enough
- Avoid inconsistency
  - Changing data once changes it everywhere

Updated Organization

Student Table

Student ID	Last Name	First Name	Department ID	email
1	Arrows	John	EE	jarrows@wam
2	Peters	Kathy	HIST	kpeters2@wam
3	Smith	Chris	HIST	smith2002@glue
4	Smith	John	CLIS	js03@wam

Department Table

Department ID	Department
EE	Electrical Engineering
HIST	History
CLIS	Information Studies

Course Table

Course ID	Course Name
lbsc690	Information Technology
ee750	Communication
hist405	American History

Enrollment Table

Student ID	Course ID	Grade
1	lbsc690	90
1	ee750	95
2	lbsc690	95
2	hist405	80
3	hist405	90
4	lbsc690	98

Updated Organization

Student Table

Student ID	Last Name	First Name	Department ID	email
1	Arrows	John	EE	jarrows@wam
2	Peters	Kathy	HIST	kpeters2@wam
3	Smith	Chris	HIST	smith2002@glue
4	Smith	John	CLIS	js03@wam

Department Table

Department ID	Department
EE	Electrical Engineering
HIST	History
CLIS	Information Studies

Course Table

Course ID	Course Name
lbsc690	Information Technology
ee750	Communication
hist405	American History

Enrollment Table

Student ID	Course ID	Grade
1	lbsc690	90
1	ee750	95
2	lbsc690	95
2	hist405	80
3	hist405	90
4	lbsc690	98

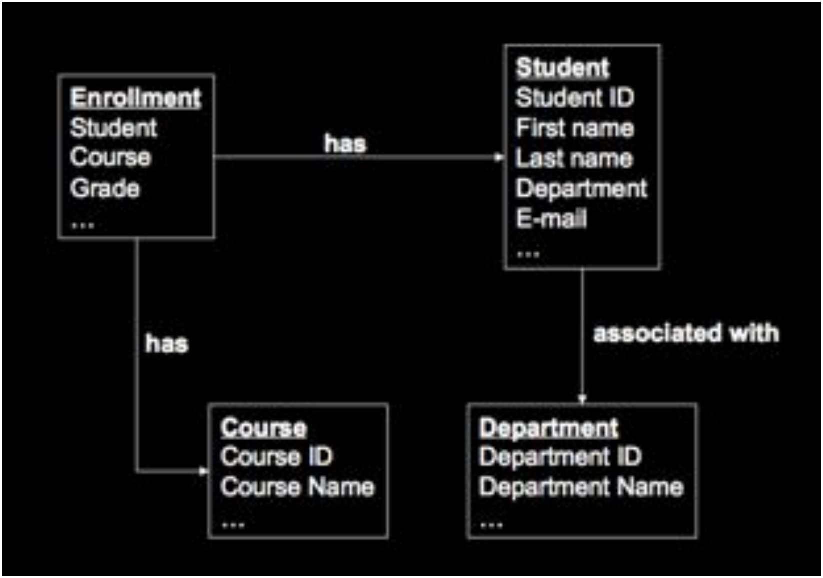
Keys

- "Primary Key" uniquely identifies a record
  - e.g., student ID in the student table
- "Foreign Key" is primary key in the <u>other</u> table
  - It need not be unique in this table

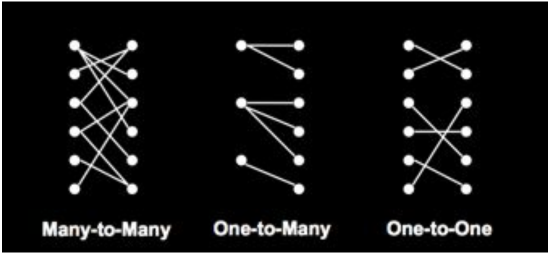
Approaches to Normalization

- For simple problems:
  - Start with the entities you're trying to model
  - Group together fields that "belong together"
  - Add keys where necessary to connect entities in different tables
- For more complicated problems:
  - Entity-relationship modeling (LBSC 670)

Entity Relationship Modeling



Entity Relationship Modeling



Database Integrity

- Registrar database must be internally consistent
  - All enrolled students must have an entry in the student table
  - All courses must have a name
  - Grades can't be negative
- What happens:
  - When a student withdraws from the university?
  - When a course is taken off the books?

Integrity Constraints

- Conditions that must be true of the database at any time
  - Specified when the database is designed
  - Checked when the database is modified
- RDBMS ensures that integrity constraints are always kept
  - So that database contents remain faithful to the real world
  - Helps avoid data entry errors
- Where do integrity constraints come from?

Outline

Relational Operations

Student Table

Student ID	Last Name	First Name	Department ID	email
1	Arrows	John	EE	jarrows@wam
2	Peters	Kathy	HIST	kpeters2@wam
3	Smith	Chris	HIST	smith2002@glue
4	Smith	John	CLIS	js03@wam

Department Table

Department ID	Department
EE	Electrical Engineering
HIST	History
CLIS	Information Studies

"Joined" Table

Student ID	Last Name	First Name	Dept ID	Department	email
1	Arrows	John	EE	Electrical Engineering	jarrows@wam
2	Peters	Kathy	HIST	History	kpeters2@wam
3	Smith	Chris	HIST	History	smith2002@glue
4	Smith	John	CLIS	Information Stuides	js03@wam

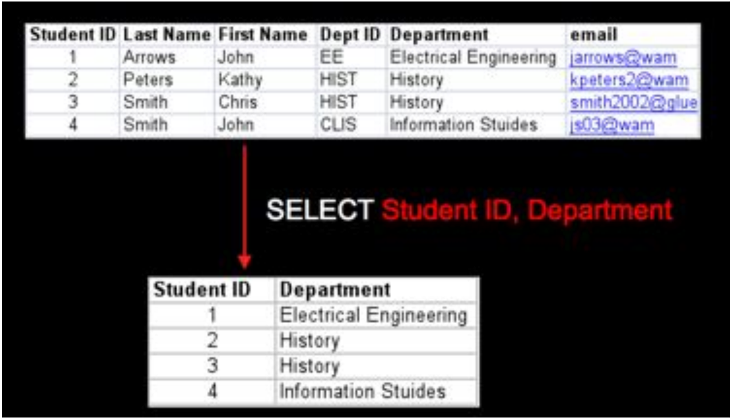
Relational Operations

Student ID	Last Name	First Name	Dept ID	Department	email
1	Arrows	John	EE	Electrical Engineering	jarrows@wam
2	Peters	Kathy	HIST	History	kpeters2@wam
3	Smith	Chris	HIST	History	smith2002@glue
4	Smith	John	CLIS	Information Stuides	js03@wam

WHERE Department ID = "HIST"

Student ID	Last Name	First Name	Department ID	Department	email
2	Peters	Kathy	HIST	History	kpeters2@wam
3	Smith	Chris	HIST	History	smith2002@glue

Relational Operations



Student ID	Last Name	First Name	Dept ID	Department	email
1	Arrows	John	EE	Electrical Engineering	jarrows@wam
2	Peters	Kathy	HIST	History	kpeters2@wam
3	Smith	Chris	HIST	History	smith2002@glue
4	Smith	John	CLIS	Information Stuides	js03@wam

Student ID	Department
1	Electrical Engineering
2	History
3	History
4	Information Stuides

Relational Operations

- Joining tables: JOIN
- Choosing columns: SELECT

Based on their labels (field names)

- Choosing rows: WHERE

Based on their contents

- These can be specified together

How is a database more than a spreadsheet?

Database in the "Real World"

- Typical database applications:
  - Banking (e.g., saving/checking accounts)
  - Trading (e.g., stocks)
  - Traveling (e.g., airline reservations)
  - Networking (e.g., Facebook)
- Characteristics:
  - Lots of data
  - Lots of concurrent operations
  - Must be fast
  - "Mission critical" (well ... sometimes)

Operational Requirements

- Must hold a lot of data
  - Use lots of computers, each with a small slice
  - So which machine has your data?
- Must be reliable
  - Use lots of computers with duplicate copies
  - How do you keep copies consistent
- Must be fast
  - Use lots of computers
  - Share the load
- Must support concurrent operations
  - This is hard
  - But often not needed

Database Transactions

- Transaction = sequence of database actions grouped together
  - e.g., transfer $500 from checking to savings
- ACID properties:
  - **Atomicity**: all-or-nothing
  - **Consistency**: each transaction must take the DB between consistent states
  - **Isolation**: concurrent transactions must appear to run in isolation
  - **Durability**: results of transactions must survive even if systems crash

Making Transactions

- Idea: keep a log (history) of all actions carried out while executing transactions
  - Before a change is made to the database, the corresponding log entry is forced to a safe location
- Recovering from a crash:
  - Effects of partially executed transactions are undone
  - Effects of committed transactions are redone
  - Trickier than it sounds!

Discussion Question

RideFinder

Design a database to match drivers with passengers (e.g., for road trips)

- Drivers post available seats; they want to know about interested passengers
- Passengers call up looking for rides: they want to know about available rides (they don't get to post "rides wanted" ads)
- These things happen in no particular order

Discussion Goals

- Design the tables you will need
  - First decide what information you need to keep track of
  - Then design tables to capture this information
- Design queries (using join, project, and restrict)
  - What happens when a passenger comes looking for a ride?
  - What happens when a driver comes to find out who his passengers are?
- Role play!

Exercise solution: tables

- **Ride**: Ride ID, Driver ID, Origin, Destination, Departure Time, Arrival Time, Available Seats
- **Passenger**: Passenger ID, Name, Address, Phone Number
- **Driver**: Driver ID, Name, Address, Phone Number
- **Booking**: Ride ID, Passenger ID

Exercise solution: queries

- Passenger calls: Can I get a ride?
  - Join: Ride, Driver
  - Project: Departure Time, Name, Phone Number
  - Restrict: Origin, Destination, Available Seats > 0
- Driver calls: Who are my passengers?
  - Join: Ride, Passenger, Booking
  - Project: Name, Phone Number
  - Restrict: (Driver) Name, Origin, Destination, Departure Time