Introduction to Data Management
CSE 344

Lecture 2: Data Models

(Ch. 2.1-2.3)

Announcements

- WQ1 and HW1 are out
  - Make sure you have signed up for Gradiance. Check Spam Folder for Signup Email!

- Use Piazza to post questions

- If you have a laptop, bring to section Th
  - also look at HW1 for installing sqlite
  - can go through the examples yourself

Announcements (cont.)

- Let us know if you have suggestions for the class
  - Don’t wait till course evals!
- Feedback to speak slower and louder
- Slides will be available before lecture (12pm)
  - We can bring hardcopies if needed

Class Overview

- Relational Data Model
  - SQL, Relational Algebra, Relational Calculus, datalog
  - Query processing and optimization
- Semistructured Data Model
  - JSon (NoSQL)
- Conceptual design
  - E/R diagrams, Views, and Database normalization
- Transactions and their implementations
- Parallel databases
  - MapReduce, and Spark

Today

- Data models
- Relational data model
- Basic SQL statements

Review

- What is a database?
  - A collection of files storing related data

- What is a DBMS?
  - An application program that allows us to manage efficiently the collection of data files

Data Models

- Suppose we have book data: author, title, publisher, pub date, price, etc
  - How should we organize such data in files?

Data model: a general, conceptual way of structuring data

What Is A Data Models

- language / notation for talking about data

- models we will use:
  - relational: data is a collection of tables
  - semi-structured: data is a tree

- other models:
  - key-value pairs: used by NoSQL systems
  - graph data model: used by RDF (semi-structured can also do)
  - object oriented: often layered on relational, J2EE

3 Elements of Data Models

- Instance
  - The actual data
- Schema
  - Describe what data is being stored
- Query language
  - How data can be retrieved and manipulated

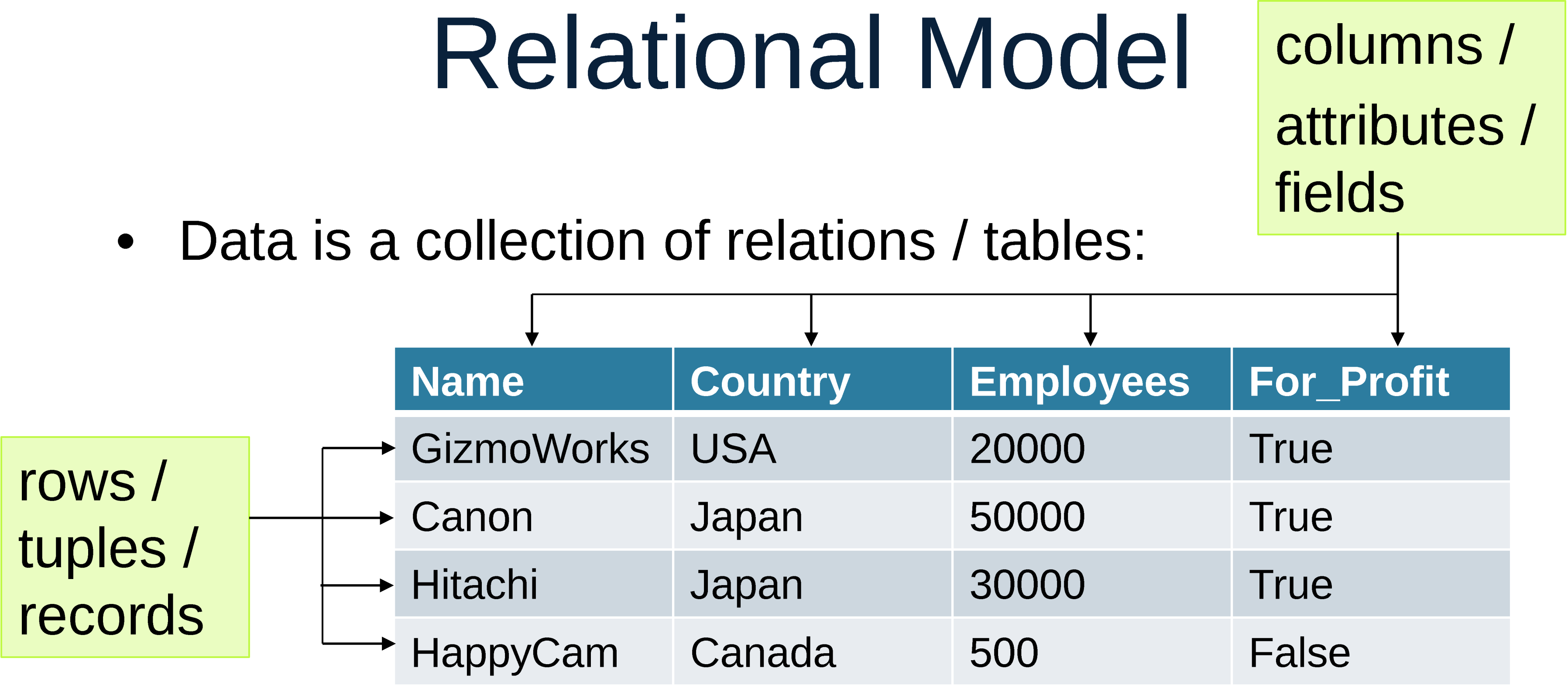
Each Data Model Does These Differently

The Relational Data Model

- Instance
  - Organized as “table” or “relation”
  - Consists of
    - “column” aka “attribute” aka “field”
    - “row” aka “tuple” aka “record”
- Schema
  - “table name” aka “relation name”
  - “column name” aka “attribute name”
  - Each attribute has a “type” aka “domain” aka “data type”

Relational Model

- Data is a collection of relations / tables:



columns / attributes / fields

rows / tuples / records

Name	Country	Employees	For_Profit
GizmoWorks	USA	20000	True
Canon	Japan	50000	True
Hitachi	Japan	30000	True
HappyCam	Canada	500	False

- mathematically, relation is a set of tuples
  - each tuple appears 0 or 1 times in the table
  - order of the rows is unspecified

The Relational Data Model

- “degree” or “arity” of a relation
  – Number of attributes

Name	Country	Employees	For_Profit
GizmoWorks	USA	20000	True
Canon	Japan	50000	True
Hitachi	Japan	30000	True
HappyCam	Canada	500	False

What is the arity of our relation?

The Relational Data Model

- Attributes must have a data type:
  - Strings: CHAR(20), VARCHAR(50), TEXT
  - Numbers: INT, SMALLINT, FLOAT
  - MONEY, DATETIME, …
  - Usually vendor specific
  - Statically and strictly enforced

In SQLite these are column “affinities”

Keys

Any
Num Lock
Caps Lock
Scroll Lock
Wake
Sleep
Power
THE ANY KEY

Keys

- An attribute that **uniquely** identifies a record.

- Is City Name A Key?
  - Seattle, Bellevue, Kathmandu, Ouagadougou

There are 41 cities in the US names Springfield

Keys

There are 41 cities in the US names Springfield

- What to do about it?

- A key can consist of multiple attributes
  - What does that mean?
  - (Springfield, ME), (Springfield, MA), (Springfield, MI), (Springfield, MN), (Springfield, M) ….

But Wisconsin has 5 Springfields!

Keys (cont.)

- A relation can have many keys
  - But only one of them can be chosen to be the *primary key*
  - DBMS often makes searches by primary key fastest
  - other keys are called “secondary”

Relation Schema

- Names and types form part of the table "**schema**":

```
Company(cname, country, no_employees, for_profit)

Company(cname: varchar(30), country: char(20),
        no_employees: int, for_profit: char(1))
```

- What is a good primary key for Company?

Relation Schema

- Names and types form part of the table "**schema**":

```
Company(cname, country, no_employees, for_profit)

Company(cname: varchar(30), country: char(20),
        no_employees: int, for_profit: char(1))
```

- Instance

cname	country	no_employees	for_profit
Canon	Japan	50000	Y
Hitachi	Japan	30000	Y

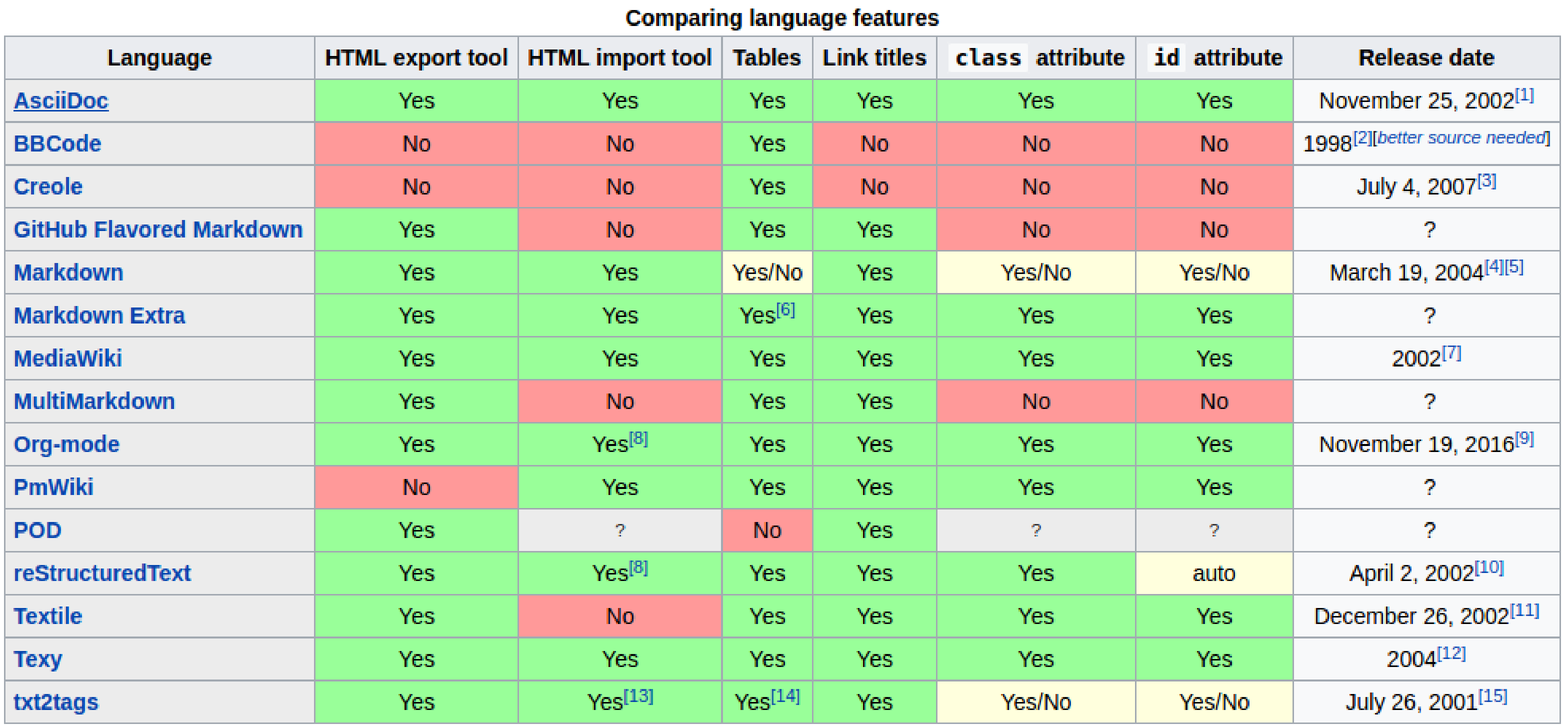
Query Language

- SQL (“sequel”)
  - **S**tructured **Q**uery **L**anguage
  - Developed by IBM in the 70s
- Declarative language

Declarative Language: Programming paradigm that expresses the logic of computation without describing its control flow

SQL is like Wiki Syntax

There are multiple standards and in practice each DBMS implements its own.

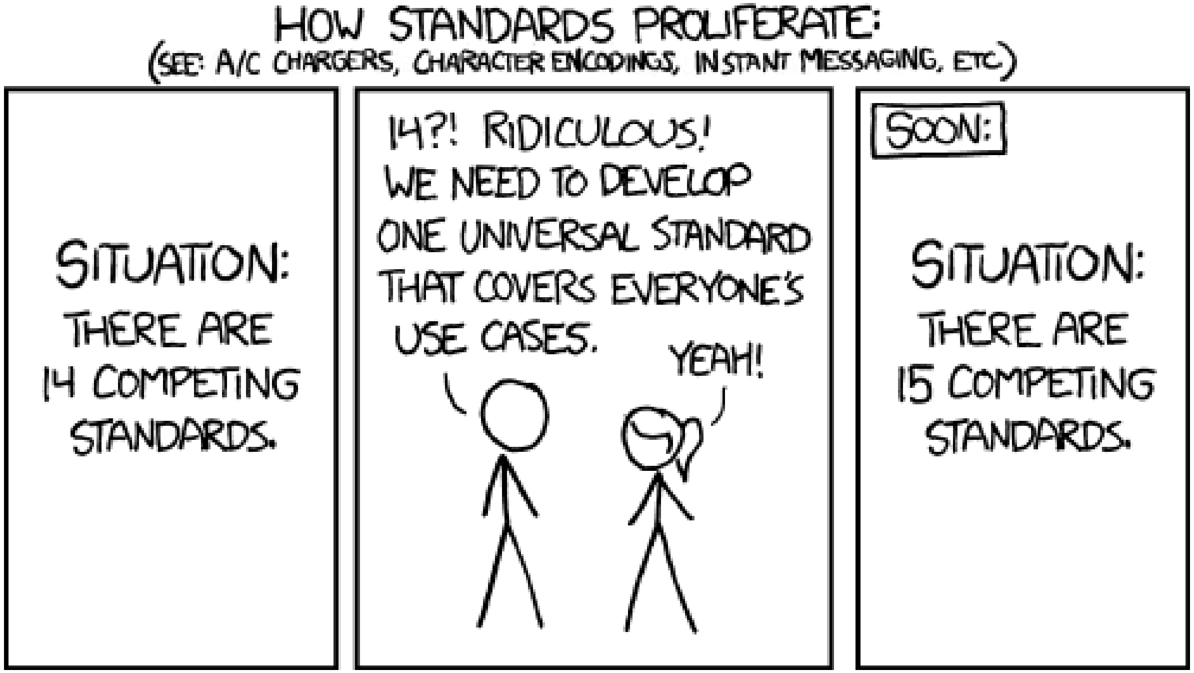


Comparing language features

Language	HTML export tool	HTML import tool	Tables	Link titles	class attribute	id attribute	Release date
AsciiDoc	Yes	Yes	Yes	Yes	Yes	Yes	November 25, 2002[1]
BBCode	No	No	Yes	No	No	No	1998[2][better source needed]
Creole	No	No	Yes	No	No	No	July 4, 2007[3]
GitHub Flavored Markdown	Yes	No	Yes	Yes	No	No	?
Markdown	Yes	Yes	Yes/No	Yes	Yes/No	Yes/No	March 19, 2004[4][5]
Markdown Extra	Yes	Yes	Yes[6]	Yes	Yes	Yes	?
MediaWiki	Yes	Yes	Yes	Yes	Yes	Yes	2002[7]
MultiMarkdown	Yes	No	Yes	Yes	No	No	?
Org-mode	Yes	Yes[8]	Yes	Yes	Yes	Yes	November 19, 2016[9]
PmWiki	No	Yes	Yes	Yes	Yes	Yes	?
POD	Yes	?	No	Yes	?	?	?
reStructuredText	Yes	Yes[8]	Yes	Yes	Yes	auto	April 2, 2002[10]
Textile	Yes	No	Yes	Yes	Yes	Yes	December 26, 2002[11]
Texy	Yes	Yes	Yes	Yes	Yes	Yes	2004[12]
txt2tags	Yes	Yes[13]	Yes[14]	Yes	Yes/No	Yes/No	July 26, 2001[15]

SQL is like Wiki Syntax

https://xkcd.com/927/


HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

SQL:1999 It introduced a large number of new features, many of which required clarifications in the subsequent SQL:2003.

Our First DBMS

- SQL Lite
- Will switch to SQL Server later in the quarter

SQL statements

- create table …
- drop table ...
- alter table ... add/remove ...
- insert into ... values ...
- delete from ... where ...
- update ... set ... where ...

See: http://www.sqlite.org/lang.html for details

Demo

Discussion

- Tables are FLAT
  - No nested attributes
- Tables DO NOT prescribe how they are implemented / stored on disk
  - This is called **physical data independence**

Physical data independence

The logical definition of the data remains unchanged, even when we make changes to the actual implementation

Adding Attributes

cname	country	no_employees	for_profit
Canon	Japan	50000	Y
Hitachi	Japan	30000	Y

- Let’s add a list of product that each company produces
  - How? Recall that tables are flat!

Foreign Keys

- A column (or columns) whose value is a key of another table
  - i.e., a reference to another row in another table

cname	country	no_employees	for_profit
Canon	Japan	50000	Y
Hitachi	Japan	30000	Y

```
Product(pname, price, category,
       manufacturer_name REFERENCES Company.cname,
       manufacture_country REFERENCES Company.country
)
```

Best Practice: Use Integer Primary Key

cid	cname	country	no_employees	for_profit
1	Canon	Japan	50000	Y
2	Hitachi	Japan	30000	Y

```
Product(pid, pname, price, category,
      manufacturer Foreign Key Company.cid)
```

pid	pname	price	category	manufacturer
1	SingleTouch	149.99	photography	1
2	AC	300	Appliance	2

Demo