LBSC 690: Session 7

Relational Databases

Jimmy Lin
College of Information Studies
University of Maryland

Monday, October 22, 2007

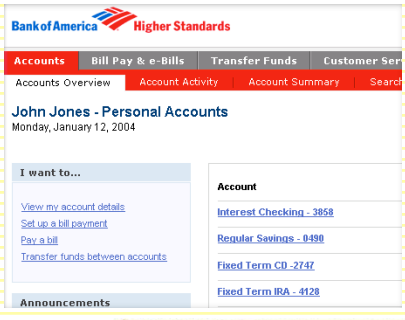
Some content borrowed from slides by Michael Franklin, UC Berkeley

Databases Then…

Databases Now…

And here…

And here…



What is a database system?

- Database:
  - a large, integrated collection of data
- Models something about the real world
  - Entities (e.g., teams, games)
  - Relationships (e.g., the Red Sox won the World Series)
  - More recently, also includes active components, often called "business logic" (e.g., the playoff system)
- A Database Management System (DBMS) is a software system designed to store, manage, and facilitate access to databases
  - Today's focus on **relational** databases

Is the WWW a DBMS? =?

- Fairly sophisticated search available
  - Crawler indexes pages on the Web
  - Keyword-based search for pages
- But, currently
  - Data is mostly *unstructured* and *untyped*
  - Can't modify the data
  - Can't get summaries, complex combinations of data
  - Few guarantees provided for freshness of data, consistency across data items, fault tolerance, …
- The picture is changing
  - New standards, e.g., XML, Semantic Web, etc., can provide richer models of data

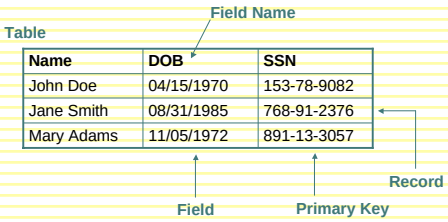
Discussion Point

- What is the difference between searching and querying?

Database Basics

- What is a database?
  - Collection of data, organized to support access
  - Models some aspects of reality
- 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 one row in the table
    - Each field is one column in the table
  - Primary Key = the field that *uniquely* identifies a record
  - Database = a collection of tables

A Simple Example



Why "Relational"?

- Databases model some aspects of reality
- A relational database views the world in terms of entities and relations between them:
  - Tables represent "relations"
  - Named fields represent "attributes"
  - Each row in the table is called a "tuple"

The Registrar Example

- What do we need to know (i.e., model)?
  - 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

A First Try

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	Information 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

Discussion: Why is this a bad idea?

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

Another Try…

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

Approaches to Normalization

- For simple problems (like the homework):
  - Start with "binary relationships": pairs of fields that are related
  - Group together wherever possible
  - Add keys where necessary
- For more complicated problems:
  - Entity relationship modeling (LBSC 670)

Some Lingo

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

The Data Model

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

Join

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

UNIVERSITY OF MARYLAND COLLEGE of INFORMATION STUDIES

Project

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

SELECT Student ID, Department

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

UNIVERSITY OF MARYLAND COLLEGE of INFORMATION STUDIES

Restrict

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

UNIVERSITY OF MARYLAND COLLEGE of INFORMATION STUDIES

Relational Operations

- Joining tables: JOIN
- Choosing columns: SELECT
  - Based on their label
- Choosing rows: WHERE
  - Based on their contents

  department ID = "HIST"
- These can be specified together

  SELECT Student ID, Dept WHERE Dept = "History"

UNIVERSITY OF MARYLAND COLLEGE of INFORMATION STUDIES

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
  - …
- What happens:
  - When a student withdraws from the university?
  - When a course is taken off the books?

UNIVERSITY OF MARYLAND COLLEGE of INFORMATION STUDIES

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?

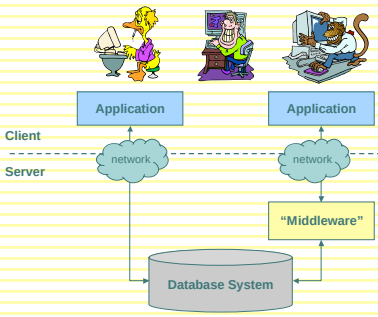
UNIVERSITY OF MARYLAND COLLEGE of INFORMATION STUDIES

Discussion Point

- How is a relational database different from a spreadsheet?

Database Architectures

two vs. three-tiered architectures



Heavy-duty Demands

- Typical database applications:
  - Banking (e.g., saving/checking accounts)
  - Trading (e.g., stocks)
  - Airline reservations
  - …
- Characteristics:
  - Lots of data
  - Lots of concurrent access
  - Must have fast access
  - "Mission critical"

Reliability

- Thought experiment: the power goes out when…
  - You're editing a file
  - You're in the middle of opening a file
  - You're in the middle of saving a file

  What happens?
- How do you build systems under such circumstances?
- *Very carefully!*

Concurrency

- Thought experiment: You and your project partner are editing the same file…
  - Scenario 1: you both save it at the same time
  - Scenario 2: you save first, but before it's done saving, your partner saves

  Whose changes survive?
  A) Yours B) Partner's C) neither D) both E) ???
- How do you build systems under such circumstances?
- *Very carefully*

Concurrency Example

- Possible actions on a checking account
  - Deposit check (read balance, write new balance)
  - Cash check (read balance, write new balance)
- Scenario:
  - Current balance: $500
  - You try to deposit a $50 check and someone tries to cash a $100 check at the same time
  - Possible sequences: (what happens in each case?)

Deposit: read balance	Deposit: read balance	Deposit: read balance
Deposit: write balance	Cash: read balance	Cash: read balance
Cash: read balance	Cash: write balance	Deposit: write balance
Cash: write balance	Deposit: write balance	Cash: write balance

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!

Entity-Relationship Diagrams

- A database models some aspect of reality…
  - ER diagrams are a way for graphically visualizing this
- Entities are captured in boxes
- Relationships are captured using arrows

The Data Model

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

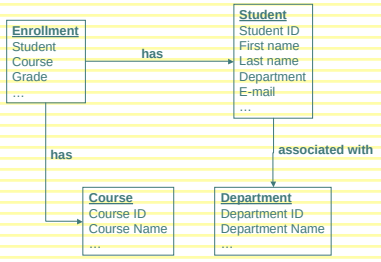
Course Table

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

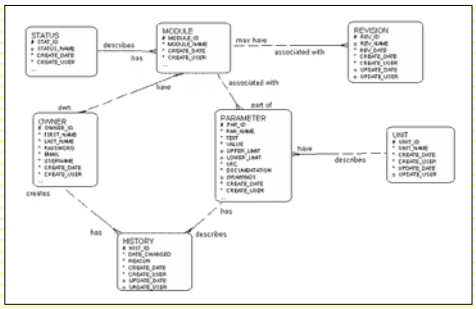
Enrollment Table

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

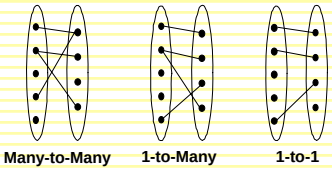
Registrar ER Diagram



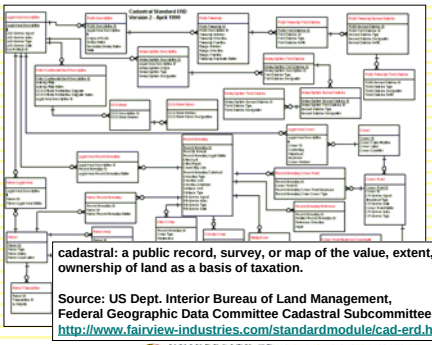
Example ER Diagram



Types of Relationships



More Complex ER Diagram



cadastral: a public record, survey, or map of the value, extent, and ownership of land as a basis of taxation.

Source: US Dept. Interior Bureau of Land Management, Federal Geographic Data Committee Cadastral Subcommittee
http://www.fairview-industries.com/standardmodule/cad-erd.htm

Steps in Database Design

- Requirements Analysis: what must database do?
- Conceptual Design: high level description (often done with ER model)
- Logical Design: translate ER into DBMS data model
- Schema Refinement: consistency, normalization
- Physical Design: indexes, disk layout
- Security Design: who accesses what, and how

RideFinder Exercise

- 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

Exercise Goals

- Identify the tables you will need
  - First decide what data you will save: What questions will be asked?
  - Then decide how to group/split it into tables
- 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?