Massachusetts Institute of Technology - Department of Urban Studies and Planning

---

11.520: A Workshop on Geographic Information Systems

11.188: Urban Planning and Social Science Laboratory

Lecture 5: Relational Databases

October 6, 2004, Joseph Ferreira, Jr.
(based on notes by Visiting Prof. Zhong-Ren Peng who taught the class in Fall 2003)

Major topics

- Database concepts and issues in GIS
- Relational Database
- Structured Query Language
- Entity-Relationship Model
- Join and Relate in ArcGIS

Limitations of one 'flat-file' for attributes

- One 'flat file' data table for each map layer is too simple a data model
  - From **lab exercise #3**: calculating percent-with-high-school
    - **cambbgrp** is read-only ==> adding a column requires building a new table
    - after calculation, must add new table and join to original **cambbgrp** table
  - From **lab exercise #3**: one-to-many issue
    - some houses in **sales89** may have sold **more than once**!
    - but, map has only **one** dot **per house**
    - what **average sale price** is correct**?** average of **most recent sale?...**
  - From **lab exercise #4**: one-to-many issue

        - Mass towns with rivers and islands have **more than one polygon**
        - Data oftern are **aggregated** at **city/town level**
        - Hence, hard to compute population density - need people / **sum**(town area)
- Strategy: **Relational Data Model** (i.e., linked tables)
    - retain tabular model for textual data **but** allow many tables that can be related to one another via common columns (attributes)
    - Allows **augmenting** attribute tables by adding additional data
    - Handles one-to-many issues (where **rows have different meaning** in joined tables)
        - create 'summary' table with one row **per town** and a column with sum of area
            - then join back via town ID to shapefile with one row **per polygon**
        - create 'summary' table with one row per house and columns for sale count, maximum price, average price, etc.
            - then join back via house ID to sales tables with one row **per sale**

- **Relational Database Management (RDBMS)** is underneath most computing
    - **database-driven web pages**: e-business catalogues, online newspapers
    - most **transaction processing**: airline reservation, ATM transaction, cellphone call logging
    - **structured query language** (**SQL**) for joining tables and specifying queries is the **lingua franca** of distributed database operations
    - Big business: Oracle, IBM (DB2), Microsoft (MS-Access, SQL-Server), ...

- Rest of Lecture
    - Intro to theory and terminology of relational database management
    - Illustrate RDBMS using MS-Access and Lab #4 datasets

A Table (relation)

Attribute name

Census_ID	Land_acre	Population	Household
10020	10.2	50	15
10021	100.5	300	80
10022	80.0	250	70

Domains: data types like integer, strings, floats, dates, etc.

The following contents of today's lecture (page 3 – 6) is derived from Longley, Goodchild, Maguire and Rhind, *Geographic Information Systems and Science*, 2001, as organized by Prof. Zhong-Ren Peng for 11.520 in Fall 2003.

Relational Databases

- A relational database is a collection of tabular *relations*, or tables.
- A *relation scheme* defines the structure of the relationship, and is composed of a set of attribute names and a mapping from each attribute name to a domain (that is, the possible values of that attribute)
- A *relation* is a finite set of tuples associated with a relation scheme in a relational database (that is, a 'table' where each row is a tuple and the columns are the things that are related)
- A *database scheme* is a set of relation schemes.

- A *relational database* is a set of relations.

A Relational database example

- Relation scheme:
  - State (State_code, state_name, state_capital)
- Database scheme:
  - State (State_code, state_name, state_capital)
  - city (city_code, City_name, State_code)
  - State_census (state, population, housing_units …)
- Key: a minimal set of attributes whose value uniquely identifies each tuple.

Operations on Relations

- *Project* operation [pick columns] applies to a single relation & returns a subset of attributes of the original.
  - SELECT census_id, population FROM census_table
  - *Restrict* operation [pick rows] works on the tuples of the table rather than the column & returns a subset of tuples of the original.
    - SELECT census_id, population FROM census_table
    - WHERE population > 0

- ***Join*** takes two relations as input and returns a single relation.
  - Join (rel1, rel2, att1, att2)

    - Joins can get complicated (drop rows if no match?, join 3+ tables, …)

- Order of operation can affect results and performance.
- Language varies slightly across vendors:
  - SELECT census_id, population, state_name
  - FROM census_table c, state_table s
  - WHERE population > 0
  - AND c.state_id = s.state_id

Structured Query Language (SQL)

- To define the database scheme (data definition),
- To insert, modify, and retrieve data from the database (data manipulation).
- Can get comples (subqueries in 'from' and 'where' clause, 3+ tables, …)
- Can be especially powerful when rows in each table have different meaning

  e.g., join parcel table to owner table, or house sales table to house table

Distributed databases and Federated databases

- Distributed databases refer to one database (or data replicates) that are distributed across multiple sites.
- Federated databases refer to many similar databases that are distributed across multiple sites but are more loosely coupled and additional rules may be needed to cross-reference tables meaningfully
- Federated databases are also called distributed relational database with fragmentations.

Conceptual data model

- To provide a data structure framework to communicate with non-specialists.
- To contain sufficient modeling constructs to capture the complexity of the system.
- To be implementation-independent.
- Two kinds of data models:
  - Entity-Relationship Model (include extended E-R model)
  - Object-Oriented Model

Entity-Relationship Model

- E-R model sees the world as inter-related entities;
- Entities (or entity types) are related with each other by a relationship.
- E-R model uses E-R diagrams to describe relations between entity types.
- E-R model describes the static state of the entity types.
- Later, we'll use MS-Access to build E-R diagram for some relational tables.

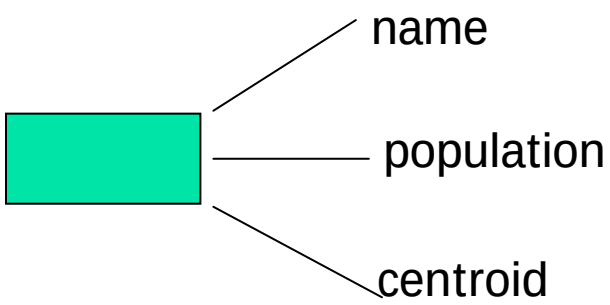
Object-Oriented Model - alternative to E-R for RDBMS

- Object-oriented model sees the world as inter-related objects.
- Object is dynamic and has its own lifespan. Hence OO model is used to deal with the dynamic nature of real-world object.
- Object = static state + functionality
- Object with similar behaviors are organized into *types*, a semantic concept.
- *Object Class* = data structure + methods, an implementation construct.

Entity and Attribute

- **Entity type**: an abstraction that represents a class of similar objects (e.g., a city, a road)
- **Entity instance**: an occurrence or instance of an *entity type* (e.g., “Cambridge” “Boston”).
- **Attribute type** serves to describe the *entity type*.
- **Attribute instance**: an occurrence of an attribute type.
- Example: City has name, population and centroid as its attribute types.
- “Cambridge” has its name, its population and its centroid as its attribute instance.

Entity type and attribute type diagram

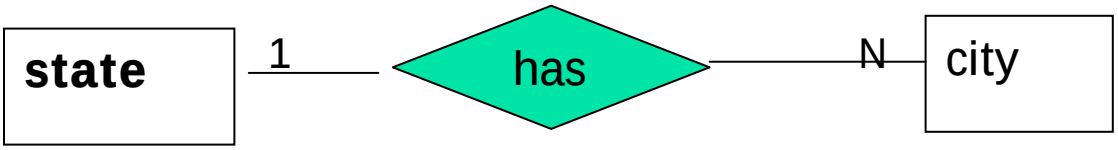


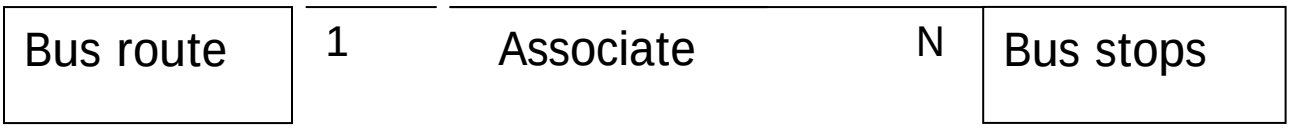
Entity-Relationship Diagrams

- One-to-One relationship

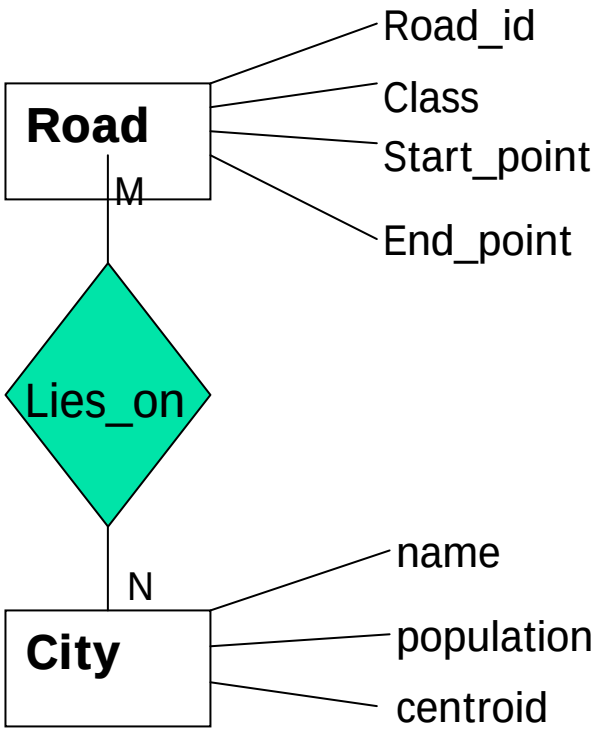
State	1	has	1	population

- One-to-Many Relationship





- Many-to-Many Relationship



Back to RDBMS and relational tables in ArcGIS

Joining and Relating Tables in ArcGIS

- Join in ArcGIS appends the attributes of the non-spatial table to the spatial (layer) attribute table.
- Relate in ArcGIS does not append attributes; only establishes a logical relationship so that when you select one record in one table you can see the matching records in the other table.
- Don't get confused between ArcGIS 'relate' (which describes the relationship between two tables, and RDBMS terminology where 'relation' = a table

When to Use Join and Relate

- Relate is preferred if the non-spatial table is maintained and updated constantly while the spatial data is not.
- Use relate when the relationship is many-to-many.
- Use relate when you have a very large non-spatial table and you don’t need all the attributes in the table.
- In other situations, you could use either.

Given the three tables below, answer the following queries

- What is the result if you associate STATE_CENSUS table with STATE table?
- What is the result if you associate STATE table with STATE_CENSUS table?
- What is the result if you associate STATE table with CITY table?
- What is the result if you associate CITY table with STATE table?

CITY table

City_code	City_name	State_code
9	Augusta	ME
10	Atlanta	GA
11	Boston	MA
12	Cambridge	MA
13	Madison	WI
14	Milwaukee	WI
15	Providence	RI

STATE table

State_Code	State_name	State_Capital
GA	Georgia	Atlanta
MA	Massachusetts	Boston
ME	Main	Augusta

STATE_CENSUS table

State	Population	Housing_units	Total area (sq miles)	Land area (sq miles)	Population/sq mile	Housing units/sq
California	33,871,648	12,214,549	163,696	155,959	217.2	78.3
Georgia	8,186,453	3,281,737	59,425	57,906	141.4	56.7
Virginia	7,078,515	2,904,192	42,774	39,594	178.8	73.3
Massachusetts	6,349,097	2,621,989	10,555	7,840	809.8	334.4
Wisconsin	5,363,675	2,321,144	65,498	54,310	98.8	42.7
Maine	1,274,923	651,901	35,385	30,862	41.3	21.1
Rhode	1,048,319	439,837	1,545	1,045	1003.2	420.9

Illustrate use of relational tables, joins, and ER diagrams

- Use MS-Access to do data manipulation for Labs #3 and #4
  - lab #4: 351 Mass cities and towns but 600+ polygons in shapefile - find density
  - Lab #3: cambbgrp.dbf: compute percent with no more than high school education

    - Lab #3: sale89_table.dbf: find homes that sold more than once; get average price
- Generate ER-diagram (table relationships)
- Understand SQL queries and graphical interface to assist in building queries

---

Last modified 6 October 2004.