T minus 4 classes

- Homework 8 and Quiz 3 are due today
- Homework 9 will be released today
  - Will be due Tuesday after Thanksgiving

CMSC 203: Lecture 23

Relations

n-ary Relations

- Subset of $A_1 \times A_2 \times \ldots \times A_n$
  - A_k is a **domain** of the relation
  - *n* is the **degree**
- Used for computer databases
  - One method is a **relational data model**

Relational Data Model

- A **database** consists of **records** which are n-tuples made up of **fields**
  - Example fields: name, number, major, GPA
  - This would be a 4-tuple
- Relations are also called **tables**
- There may be a **primary key** – a domain that unique identifies an n-tuple
- A **composite key** is when a combination of domains are used to uniquely identify an n-tuple

Operations

- Operations on n-ary relations form new n-ary relations
- We can “query” on databases
  - Typically satisfy certain conditions or criteria
- Include:
  - selection
  - projection
  - join

Selection

- Let *R* be an n-ary relation and *C* be a condition
- **selection** s_C maps R to all n-tuples in R that satisfy C
  - Example:
    - C_1 is condition Major = “Computer Science”
    - C_2 is condition GPA > 3.5
    - C_3 is $(C_1 \wedge C_2)$
    - Apply s_{C3} on a table of all students for a table of CMSC majors with GPA above 3.5

Selection Example

Name	ID	Major	GPA
Ackermann	231455	Computer Science	3.88
Adams	888323	Physics	3.45
Chou	102147	Computer Science	3.49
Rao	678543	Mathematics	3.90
Stevens	786576	Computer Science	3.52

$\downarrow S_{C3}$

Name	ID	Major	GPA
Ackermann	231455	Computer Science	3.88
Stevens	786576	Computer Science	3.52

Projection

- **projection** $P_{i_1 i_2 \ldots i_m}$ where $i_1 < i_2 < \ldots < i_m$

  maps n-tuple $(a_1, a_2, \ldots, a_n)$ to m-tuple $(a_{i_1}, a_{i_2}, \ldots, a_{i_m})$
- Essentially deletes n – m components of n-tuple
- *Example* : $P_{1,4}$ may be used to only keep first and fourth fields in a database
- May result in fewer rows if tuples are not unique

Projection Example

Name	ID	Major	GPA
Ackerman	231455	CMSC	3.88
Adams	888323	Physics	3.45
Chou	102147	CMSC	3.49
Rao	678543	Math	3.90
Stevens	786576	CMSC	3.52

$\xrightarrow{P_{1,4}}$

Name	GPA
Ackerman	3.88
Adams	3.45
Chou	3.49
Rao	3.90
Stevens	3.52

Projection Example

Name	Major	Course
Glauser	CMSC	CMSC 202
Glauser	CMSC	CMSC 203
Glauser	CMSC	DANCE 101
Miller	MATH	MATH 190
Miller	MATH	MATH 192

$$\xrightarrow{P_{1,2}}$$

Name	Major
Glauser	CMSC
Miller	MATH

Join

- *R* is a relation of degree *m* and *S* is relation of degree *n*
- **join** $J_p(R,S)$ is relation of degree m+n-p which consists of all $(a_1,a_2,\ldots,a_{m-p},c_1,c_2,\ldots,c_p,b_1,b_2,\ldots,b_{n-p})$ where
  - m-tuple $(a_1,a_2,\ldots,a_{m-p},c_1,c_2,\ldots,c_p)$ belongs to R and
  - n-tuple $(c_1,c_2,\ldots,c_p,b_1,b_2,\ldots,b_{n-p})$ belongs to S
- Combines all tuples between two relations by matching where elements agree

Join Example

Professor	Department	Course
Farber	Psychology	501
Farber	Psychology	617
Grammer	Physics	544
Grammer	Physics	551
Rosen	Comp. Sci.	518
Rosen	Math	575

Department	Course	Room	Time
Psychology	501	A100	3:00 PM
Psychology	617	A110	11:00 AM
Physics	544	B505	4:00 PM
Comp. Sci.	518	N521	2:00 PM
Math	575	N502	3:00 PM
Math	611	N521	4:00 PM

J_2

Professor	Department	Course	Room	Time
Farber	Psychology	501	A100	3:00 PM
Farber	Psychology	617	A110	11:00 AM
Grammer	Physics	544	B505	4:00 PM
Rosen	Comp. Sci.	518	N521	2:00 PM
Rosen	Math	575	N502	3:00 PM

SQL

- **Structured Query Language** can carry out operations on our *n*-ary relations
  - Uses terms introduced in beginning of section
- *Examples* :
  - `SELECT Departure_time FROM Flights WHERE Destination='Detroit'`
  - `Select Professor, Time FROM Teaching_assignments, Class_schedule WHERE Department='Mathmatics'`

Summary of n-ary Relations

- You will be expected to:
  - Understand the terms presented at the beginning
  - Select a suitable primary / composite key
  - Perform operations on n-ary relations
- You will NOT be expected to:
  - Know SQL

Relations as Matrices

- Represent relation between finite sets in computer programs using **zero-one matrix**
  - Elements must be in particular, but arbitrary, order
- $$M_R = [m_{ij}] \text{ where } m_{ij} = \begin{cases} 1 & (a_i, b_j) \in R \\ 0 & (a_i, b_j) \notin R \end{cases}$$
- Basically, 1 if a_i is related to b_j, 0 otherwise
- Depends on the ordering

Example of Creating Relation Matrix

- Suppose that A = {1, 2, 3} and B = {1, 2}
- R is relation from A to B containing (a, b) if a > b

$$M_R = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}$$

Example of Deriving Relation

- Which ordered pairs are in *R* given the matrix below?

$$M_R = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

-

Example of Deriving Relation

- Which ordered pairs are in *R* given the matrix below?

$$M_R = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

- $R = \{(a_1, b_2), (a_2, b_1), (a_2, b_3), (a_2, b_4), (a_3, b_1), (a_3, b_3), (a_3, b_5)\}$

Properties of Relations as Matrices

- **Reflexive** iff $m_{ii} = 1$ for $i = 1, 2, \ldots, n$
- **Symmetric** iff $m_{ji} = m_{ij}$ for all i and j
- **Antisymmetric** iff $m_{ij} = 0$ or $m_{ji} = 0$ when $i \neq j$
- Example: What are the properties of M_R?

$$M_R = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

Digraphs

- Each element is represented as a point
- Each ordered pair is a directional arrow
- Referred to as a **directed graph** (aka: **digraph**)

Digraph Definition

- Set *V* of **vertices** (**nodes**)
  - Elements in domain
- Set *E* of **edges** *(***arcs**)*:*
  - Ordered pairs of elements in V
- Given edge (a, b)
  - *a* is the **initial vertex**
  - *b* is the **terminal vertex**
- Edge (a, a) uses arch called a **loop**

Properties of Relations as Digraphs

- **Reflexive** iff loop at every vertex
- **Symmetric** iff every edge between distinct vertices has edge in opposite direction
  - May also be represented using undirected graphs
- **Antisymmetric** iff never two edges in opposite directions between distinct vertices
- **Transitive** iff edge from x to y and edge from y to z, there is an edge from x to z