* **12/11/17**
* Relationships of Higher degree
  + Degree 2 are binary
  + Degree 3 are ternary
  + Degree n are n-ary
  + Constraints are harder for n>2
* Entities - nouns
* Relationships - verbs
* **12/13/17**
* Liveliness and Safety
  + R(SRS) - required by SRS
  + F(SRS) - Forbidden by SRS
  + R(M): Liveliness criteria is required behavior of system model (M)
  + F(M): safety criteria is the forbidden behavior of the system model (M)
* Analyzing problem
  + FInd Required and forbidden conditions
  + ID entities, attributes, and relations
  + Design complete ER
  + Convert ER model to Relational model
* **01/03/18**
* Relational Model
  + Relation Tables
  + Based on ER
  + R(A, B, C) //relational model with 3 atributes
    - Domain of an attribute corresponds to a set of values an attribute can take
    - dom(A) = {1, 2, 3}
    - dom(B)= set of all strings of length 25
    - dom(C) = {‘csc’, ‘bme’, ‘cyen’}
  + Definition: Relation - A relation ‘R’ based of atributes A&B can be defined as
    - R (all possible combinations) dom(A) x dom(B)
  + Definition: Relational Algebra - RA is a formal query language based on a set of operations over Relations
  + Fundamental Operators (set theory)
    - Select
    - Project
    - Cartesian Product
    - Union
    - Set difference
    - Rename
  + Additional Operators
    - Natural Join
    - Intersection
    - Division
    - 0-Join
  + Select operation (σ)
    - Selects a subset of tuples
    - Unary operation
  + Project Operator (𝛱)
    - Select a subset of attributes
    - Unary
    - If duplicate entries created, it will remove them
  + Cartesian product (X)
* **01/05/18**
* Relationships and constraints
  + Structural Constraints
    - Cardinality Ratio [maximum]
    - Participation/existential Constraints [minimum]
* #1 (1:N) relationship
  + Requirement: every employee works for a department; and a department can have many employees. A new department need not have any employees
  + 1 Degree of the relationship
    - num of entities in the relationship: (2)
  + 2 Cardinality ratio
    - max number of relationships that an instance of the entity can participate in: emp(1) Dept(n)
  + 3 Minimum Emp(1) Dept(0)

|  |  |  |
| --- | --- | --- |
|  | Emp | Dept |
| Max | 1 | N |
| Min | 1 | 0 |

* + Cardinality rep

[emp]===========<Works for>-----------[Dept]

N 1

* + Min-max rep

[emp]-----------<works for>-----------[Dept]

(1,1) (0,N)

* #2 (1:1) relationship
  + Every Dpartment must have a manager, and only one employee manages a department, an employee can manage only one department

[Emp]---------<manages>--------[Dept]

(0,1) (1,1)

* + 1 Degree: 2
  + 2 Cardinality(max) Emp(1) Dept(1)
  + 3 Minimum Emp(0) Dept(1)
* #3 (N:M) Every employee has to work on at least 1 project; every project should have at least 1 employee
  + 1 degree: 2
  + 2 Max emp(n) proj(M)
  + 3 min emp(1) proj(1)

[emp]------<works on>-----(proj)

(1,N) (1,M)

* #4 Recursive Relationship
  + Every employee is supervised by a supervisor who is an employee
    - except for CEO, has no boss
      * Would raise min for supervisee to 1 if CEO had boss
  + (still use 3 ovals, just add designation of supervisor/supervise)
  + 1 Degree 2
  + 2 Max emp(1) sup(N)
  + 3 Min emp(0) sup(0)

-<supervise>-

| |

| |

----<emp>----

(0,N) (0,1)

* **1/10/18**
* Operators in RA
  + Set Difference (-)
    - Binary Operator
    - r&s -> relations
    - r - s
      * Tuples in r, that are not in s
  + Set Union(U)
    - Binary Operator
    - r&s -> relations
    - r U s
      * All tuples in both r&s that are unique and should have common attributes
  + Set Intersection (**∩)**
  + Natural Join (⋈)
    - Case 1
    - Case 2
  + 𝛳-join
  + ~~Division or quotient~~
* RA in SQL queries
* Examples
  + Book(AccNo, YrPub, Title)
  + Borrow(AccNo, CdNo, DOI)
  + User(CdNo, BName, BAdd)
  + Supplier(SName, SAdd)
  + Supply(AccNo, CdNo, DOI)
  + R-S
    - 𝛱AccNo(Book) - 𝛱AccNo(Borrow)
      * Find all titles of books not borrowed
  + RUS
    - 𝛱AccNo(supply) U 𝛱AccNo(borrow)

Select 3

From 1

Where 2

Pi(attributes) (sigma(terms)(place (bowtie) place))

Project ideas:

* food pantry management
* disaster shelter management
* traffic ticket management
* mobile medical data management (ambulances/fire)
* student university life management????? (dorm??)
* **Grad Students**
* Data stream management systems
* probabilistic data systems
* rule based multi-query optimization