2. Data Model (2/2)*

Relational Calculus

- Relational Algebra needs to specify the order of operations; while relational calculus only needs to indicate the logic condition the result must be fulfilled.
- Comes in two flavors: *Tuple relational calculus* (TRC) and *Domain relational calculus* (DRC).
- Calculus has *variables, constants, comparison ops, logical connectives* and *quantifiers*.
  - *TRC*: Variables range over (i.e., get bound to) *tuples*.
  - *DRC*: Variables range over *domain elements* (attribute values).
  - Both TRC and DRC are simple subsets of first-order logic.
- Expressions in the calculus are called *formulas*. An answer tuple is essentially an assignment of constants to variables that make the formula evaluate to *true*.

Domain Relational Calculus

- *Query* has the form:

  $\{ \langle x_1, x_2, \ldots, x_n \rangle \mid P(x_1, x_2, \ldots, x_n, x_{n+1}, \ldots, x_{n+m}) \}$

- $x_1, x_2, \ldots, x_n, x_{n+1}, \ldots, x_{n+m}$ are called *domain variables*. $x_1, x_2, \ldots, x_n$ appear in result.

- ✓ *Answer* includes all tuples $\langle x_1, x_2, \ldots, x_n \rangle$ that make the *formula* $P(x_1, x_2, \ldots, x_n, x_{n+1}, \ldots, x_{n+m})$ be *true*.

- ✓ *Formula* is recursively defined, starting with simple *atomic formulas* (getting tuples from relations or making comparisons of values), and building bigger and better formulas using the *logical connectives*.

DRC Formulas

- *Atomic formula:*
  - $\langle x_1, x_2, \ldots, x_n \rangle \in Rname$, or X *op* Y, or X *op* constant
  - *op* is one of $<, >, =, \leq, \geq, \neq$
- *Formula:*
  - an atomic formula, or
  - $\neg p$, $p \wedge q$, $p \vee q$, where p and q are formulas, or
  - $\exists X(p(X))$, where variable X is *free* in p(X), or
  - $\forall X(p(X))$, where variable X is *free* in p(X)

Free and Bound Variables

- The use of quantifiers $\exists X$ and $\forall X$ is said to *bind* X.
  - A variable that is not bound is free.
- Let us revisit the definition of a query:

  $<x_1,x_2, \ldots, x_n> \mid P(x_1,x_2,\ldots,x_n,x_{n+1},\ldots, x_{n+m})$
- There is an important restriction: the variables $x_1,x_2, \ldots, x_n$ that appear to the left of ' | ' must be the *only* free variables in the formula p(...).

Find all sailors with a rating above 7

- $\{ \langle I,N,T,A \rangle \mid \langle I,N,T,A \rangle \in Sailors \wedge T>7 \}$
- The condition $\langle I,N,T,A \rangle \in Sailors$ ensures that the domain variables *I*, *N*, *T* and *A* are bound to fields of the same Sailors tuple.
- The term $\langle I,N,T,A \rangle$ to the left of ' |' (which should be read as *such that*) says that every tuple $\langle I,N,T,A \rangle$ that satisfies $T>7$ is in the answer.
- Modify this query to answer:
  - Find sailors who are older than 18 or have a rating under 9, and are called 'Joe'.

Unsafe Queries, Expressive Power

- It is possible to write syntactically correct calculus queries that have an infinite number of answers! Such queries are called *unsafe*.
  - e.g., {S | ¬(S ∈ Sailors)}
- It is known that every query that can be expressed in relational algebra can be expressed as a safe query in DRC / TRC; the converse is also true.
- *Relational Completeness*: {**σ, π, ∪, −, ×**} is a complete operation set. Relational calculus can express these five operations easily, so relational calculus is also *Relational Completeness*. SQL language is based on relational calculus, so it can express any query that is expressible in relational algebra /calculus.

Tuple Relational Calculus

- *Query* has the form:

  { t[<*attribute list*>] | P(t)}

- t is called *tuple variable*.

- ✓ *Answer* includes all tuples t<*attribute list*> that make the *formula* P(t) be *true*.

- ✓ Example query: Find all sailors' name whose rating above 7 and younger than 50;

  { t[N] | t ∈ *Sailors* ∧ t.T>7 ∧ t.A<50 }

Remarks to Traditional Data Model

- Hierarchical, Network, and Relational Model
- Suitable for OLTP applications
- Based on record, can't orient to users or applications better
- Can't express the relationships between entities in a natural mode.
- Lack of semantic information
- Few data type, hard to fulfill the requirements of applications

2.4 ER Data Model

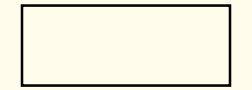
- *Entity:* Real-world object distinguishable from other objects. An entity is described (in DB) using a set of *attributes*.
- *Entity Set*: A collection of similar entities. E.g., all employees.
  - All entities in an entity set have the same set of attributes. (Until we consider ISA hierarchies, anyway!)
  - Each entity set has a *key*.
  - Each attribute has a *domain*.
  - Permit combined or multi-valued attribute

Relationship

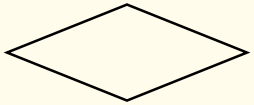
- *Relationship*: Association among two or more entities. E.g., Attishoo *works in* Pharmacy department.
  - Relationship can have attributes
- *Relationship Set*: Collection of similar relationships.
  - An n-ary relationship set R relates n entity sets E_1 ... E_n; each relationship in R involves entities e_1, ..., e_n
  - Same entity set could participate in different relationship sets, or in different "roles" in same set.

ER Diagram

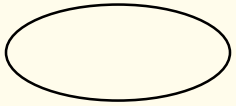
- Concept model: entity—relationship, be independent of practical DBMS.
- Legend:



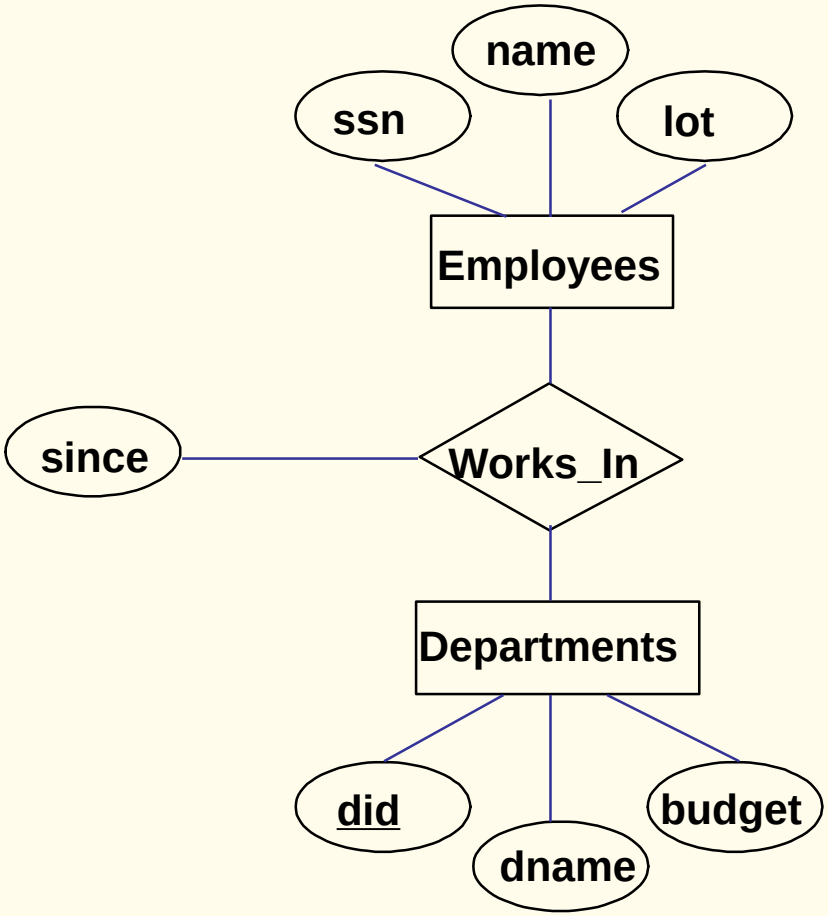
entity



relation

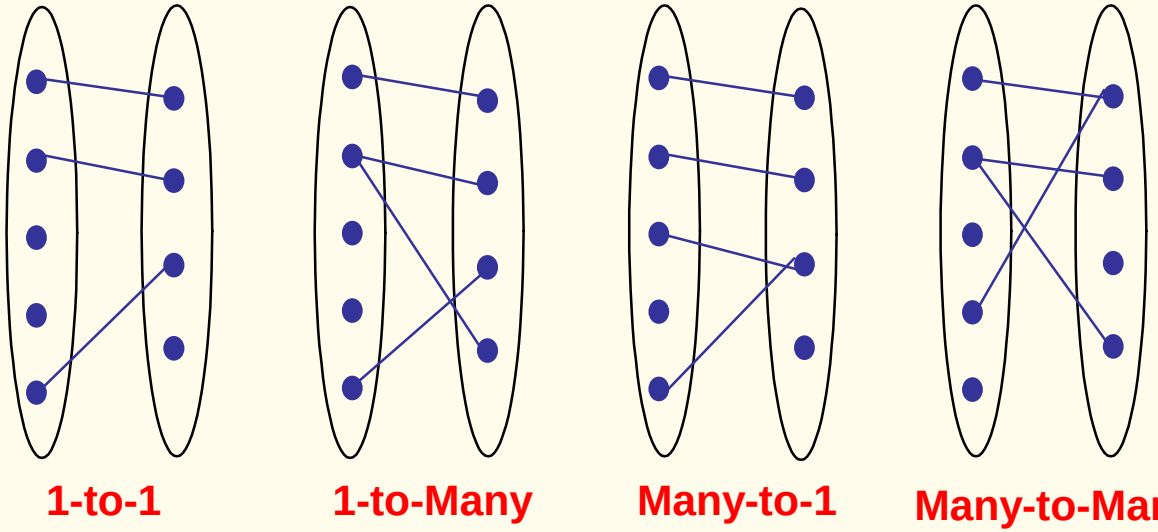


attribute



Cardinality Ratio Constraints

- Relationships can be distinguished as 1:1, 1:N, and M:N. This is called cardinality ratio constraints.



- For example: an employee can work in many departments; a dept can have many employees. This M:N. In contrast, each dept has at most one manager and one employee can only be manager of one dept, then this is 1:1.

Participation Constraints

- We can further specify the minimal and max number an entity participates a relationship. This is called participation constraints.
- If a department must have a manager, then we say the Departments is *total participation* in Manages relationship (*vs. partial*). The minimal participating degree of Departments is 1.
- Another example: in the selected course relationship between Students and Courses, if we specify every student must select at least 3 courses and at most 6 courses, the participating degree of Students is said to be (3,6).

Advanced Topics of ER Model

- Weak entity
- Specialization and Generalization
  - Similar as inheriting in Object-Oriented data model
- Aggregation
  - Allows us to treat a relationship set as an entity set for purposes of participation in (other) relationships
- Category
  - Allow us to express an entity set consists of different types of entities. That is, a hybrid entity set.

2.5 Object-Oriented Data Model

- The shortage of relational data model
- Break through 1NF
- Object-Oriented analysis and programming
- Requirement of objects' permanent store
- Object-Relation DBMS
- Native (pure) Object-Oriented DBMS

2.6 Other Data Models

- Logic-based data model (Deductive DBMS)
  - Extend the query function of DBMS (especially recursive query function)
  - Promote the deductive ability of DBMS
- Temporal data model
- Spatial data model
- XML data model
  - Store data on internet
  - Common data exchange standard
  - Information systems integration
  - Expression of semi-structured data
  - … …
- Others

2.7 Summary

- Data model is the core of a DBMS
- A data model is a methodology to simulate real world in database
- In fact, every kind of DBMS has implemented a data model

☹ If there will be a data model which can substitute relational model and become popular data model, just as relational model substituted hierarchical and network model 30 years ago **???**