

Session 2: E-R Data Model (CH-2)
CSCI-585, Cyrus Shahabi

- Data Model: A collection of conceptual tools for describing data, data relationships, data semantics & consistency constraints. Example: Entity Relationships Data Model (E-R model)

- Based on a perception of a real world which consists of a set of basic objects called entities and relationships among these objects.

- Entity: An entity is an object that exists and is distinguishable from other objects (e.g., Tony with SS#, csci585 in Spring-2001, …)

- Entity set: A set of entities of the same type (e.g., students, courses). Presented as:

- Entity sets need not to be disjoint (e.g., Pamu (TA) is both a student and an employee of USC)

- Attributes: An entity is represented by a set of attributes (e.g., student: name, SS#, age, address, …). Attribute can be considered as a function that maps an entity into a domain (e.g., SS#: entity->integer). Presented as:

- Relationship: A relationship is an association among several entities (e.g., enrolled relationship associates Tony with csci-585)

- Relationship set: A set of relationships with the same type. Formal definition:

- Role: The function that an entity plays in a relationship is called its role. Normally implicit.

- Recursive relationship: Same entity set participates more than once in a relationship in different roles. Role names, hence, become essential:
• Mapping cardinalities (example of definition of constraints in a data model): The number of entities to which another entity can be associated via a relationship set (depends on the real-world that is being modeled by the relationship set).

• The following is possible mapping cardinality of relation R between two entity sets A and B (binary):
  • One-to-one (1:1): Women marrying Men (assuming no polygamy!)
  • Many-to-one (N:1): Children having mothers
  • one-to-many (1:N): Mothers having children
  • many-to-many (M:N): Students enrolled in courses

• A relationship may also have descriptive attributes:

(A relationship must be uniquely identified by the participating entities, without reference to descriptive attributes)

• Keys: Entities and relationships are distinguishable using various keys.
• Superkey: A combination of one or more attributes that allow us to identify uniquely an entity in an entity set (e.g., SS#, name & SS#).
• Candidate key: A minimal superkey (no proper subset is a superkey) that uniquely identifies an entity (e.g., SS#, name & address, phone#).
• Primary key: A candidate key chosen by DBA to identify entities of an entity set (e.g., SS#).

• Degree of relationship: Number of participating entity sets in a relationship.
  • Binary relationships: A relationship that involves two entities (e.g., enrolled)
  • Ternary (N-ary) relationships: A relationship that involves three (N) entities (e.g., Tony enrolled in csci-585 at USC)

• It is always possible to replace a non-binary relationship set by a number of distinct binary relationship sets (e.g., Tony enrolled in csci-585, csci-585 is-offered at USC). However, depending on the actual real-world problem, one may be more appropriate than the other.

• Weak entity set: An entity set that does not have enough attributes to form a primary key (e.g., transaction#, date, amount. Different accounts might have similar transaction#).

• Strong entity set: One with a primary key.
• How to distinguish entities of a weak entity set?
• Discriminator: Set of attributes in a weak entity set that allow distinguishing among all those entities in the entity set that depends on one particular strong entity (e.g. transaction# is unique within the same account#)
• Primary key of a weak entity set is formed by the primary key of the strong entity set on which it is existence-dependent (termed, owner entity set), plus it’s discriminator (e.g., account#+transaction#).
• Attributes for relationships: Can be migrated to one of the participating entity sets for 1:1, 1:N and N:1 relationships (which one?). But NOT for M:N relationships.
• Attribute types:
  • **Composite** vs. simple: Useful when we sometimes need to refer to the entire attributes as a unit and sometimes to each of components:

![Composite Attributes Diagram]

  - Street
  - City
  - State
  - Zipcode
  - Address

  - Degrees

• **Multivalued** vs. single-valued: An attribute with a set of values for a same entity.

  - Age

  - Degrees

• **Derived** vs. stored: The value of the attribute can be derived from either other attributes (e.g., age from DOB) or related entities (e.g., NumberOfEmployees).

• **Null** attribute: When a value is *not applicable* for an attribute of a particular entity (e.g., AppartmentNumber, Degrees); or the value exists but is *missing* (e.g., null value for Weight of a person); or the value is *not known* to exist or not (e.g., null value for phone#).

(An interactive/attractive example 😊 here)