Constraints in Entity-Relationship Models

Zaki Malik

September 18, 2008
Types of Constraints

- **Keys** are attributes or sets of attributes that uniquely identify an entity within its entity set.
- **Single-value constraints** require that a value be unique in certain contexts.
- **Referential integrity constrains** require that a value referred to actually exists in the database.
- **Domain constraints** specify what set of values an attribute can take.
- **General constraints** are arbitrary constraints that should hold in the database. We will study some examples in the second half of the semester.
- **Constraints are part of the schema of a database.**
Keys

• A *key* is a set of attributes for one entity set such that no two entities in this set agree on all the attributes of the key.
  – It is allowed for two entities to agree on some, but not all, of the key attributes.

• A key for an entity set E is a set K of one or more attributes such that given any two entities e1 and e2 in E, e1 and e2 cannot have identical values for all the attributes in K.

• E can have multiple keys. We usually designate one as the primary key.

• We **must** designate a key for every entity set.
Keys in E/R Diagrams

• Underline the key attribute(s).
• In an Isa hierarchy, only the root entity set has a key, and it must serve as the key for all entities in the hierarchy.
Example: a Multi-attribute Key

- Note that *hours* and *room* could also serve as a key, but we must select only one key.
Examples of Keys
Single-Value Constraint

• There is at most one value in a given context.

1. Each attribute of an entity set has a single value.
   – If the value is missing, we can invent a “null” value.
   – E/R models cannot represent the requirement that an attribute cannot have a null value.

2. A many-one relationship implies a single value constraint.
Referential Integrity Constraint

• Asserts that exactly one value exists in a given context.
• Usually used in the context of relationships.
• Example: Many-one Advises relationship between Students and Professors.
  – Many-one requirement says that no student may have more than one advising professor.
  – Referential integrity constraint says that each student must have exactly one advising professor and that professor must be present in the database.
• If R is a (many-to-one or one-to-one) relationship from E to F, we use a rounded arrowhead pointing to F to indicate that we require that the entity in F related by R to an entity in E must exist.
Example

• Each department has at most one chairperson who is its head (there are times when a department may not have a chairperson).

• Each chairperson can be the head of at most one department and this department must exist in the database.

• Where do we put the arrows?
Enforcing Referential Integrity Constraints

- We forbid the deletion of a referenced entity (e.g., a professor) until the professor advises no students.
- We require that if we delete a referenced entity, we delete all entities that reference it.
- When we insert a student entity, we must specify an existing professor entity connected to the student by the Advises relationship.
Weak Entity Sets

• Occasionally, entities of an entity set need “help” to identify them uniquely.

• Entity set $E$ is said to be weak if in order to identify entities of $E$ uniquely, we need to follow one or more many-one relationships from $E$ and include the key of the related entities from the connected entity sets.
Example

• *name* is almost a key for football players, but there might be two with the same name.
• *number* is certainly not a key, since players on two teams could have the same number.
• But *number*, together with the *Team* related to the player by *Plays-on* should be unique.

- Double diamond for *supporting* many-one relationship.
- Double rectangle for the weak entity set.
Weak Entity-Set Rules

• A weak entity set has one or more many-one relationships to other (supporting) entity sets.
  – Not every many-one relationship from a weak entity set need be supporting.

• The key for a weak entity set is its own underlined attributes and the keys for the supporting entity sets.
  – E.g., player-number and team-name is a key for Players in the previous example.
Example of Weak Entity Set

- Each department teaches multiple courses. Each course has a number. What is the key for the entity set Courses?
Design Techniques

• Be faithful to the specification of the application.
• Avoid redundancy.
• Keep the entities and relationship simple.
  – Don’t use an entity set when an attribute will do.
• Select the right relationships.
• Select the right type of element.
• Limit the use of weak entity sets.
Be Faithful

• Do not use meaningless or unnecessary attributes.

• Define the multiplicity of a relationship appropriately.
  – What is the multiplicity of the relationship Take between Students and Courses?
  – What is the multiplicity of the relationship Teach between Professors and Courses?
Avoiding Redundancy

• Redundancy occurs when we say the same thing in two different ways.

• Redundancy wastes space and (more importantly) encourages inconsistency.
  – The two instances of the same fact may become inconsistent if we change one and forget to change the other, related version.
Example: Bad

This design states the manufacturer of a beer twice: as an attribute and as a related entity.
Example: Bad

This design repeats the manufacturer’s address once for each beer; loses the address if there are temporarily no beers for a manufacturer.
Example: Good

This design gives the address of each manufacturer exactly once.
Entity Sets Versus Attributes

• An entity set should satisfy at least one of the following conditions:
  – It is more than the name of something; it has at least one nonkey attribute.
  or
  – It is the “many” in a many-one or many-many relationship.
Example: Good

- *Manfs* deserves to be an entity set because of the nonkey attribute *addr*.
- *Beers* deserves to be an entity set because it is the “many” of the many-one relationship *ManfBy*. 
Example: Bad

Since the manufacturer is nothing but a name, and is not at the “many” end of any relationship, it should not be an entity set.
Example: Good

There is no need to make the manufacturer an entity set, because we record nothing about manufacturers besides their name.
Design

- Do not add unnecessary relationships.
- It may be possible to deduce one relationship from another.
- Do we need the relationship Instruct between Professors and Students?
  - No. We can deduce this relationship from Take and Teach.
Design

• Do not add unnecessary relationships.
• It may be possible to deduce one relationship from another.
• Do we need the relationships Take and Teach?
  – Yes. Why?
Select the Right Type of Element

- Attribute or Entity or Relationship?
- Can we make Professor an attribute of Courses and remove the relationship Teach?
- What if we add the relationship Evaluation?
- What if we add the relationship Research signifying a research project the student is working on with a professor?
Converting an Entity Set into an Attribute

- If an entity set E satisfies the following properties:
  1. All relationships involving E have arrows entering E.
  2. The attributes of E collectively identify an entity (i.e., no attribute depends on another).
  3. No relationship involves E more than once
- then we can replace E as follows:
  1. If there is a many-one relationship R from an entity set F to E, remove R and make the attributes of E be attributes of F.
  2. If there is a multi-way relationship R with an arrow to E, make the attributes of E be new attributes of R and remove the arrow from R to E.
Don’t Overuse Weak Entity Sets

• Beginning database designers often doubt that anything could be a key by itself.
  – They make all entity sets weak, supported by all other entity sets to which they are linked.

• In reality, we usually create unique ID’s for entity sets.
  – Examples include social-security numbers, automobile VIN’s etc.
When Do We Need Weak Entity Sets?

- The usual reason is that there is no global authority capable of creating unique ID’s.
- Example: it is unlikely that there could be an agreement to assign unique player numbers across all football teams in the world.
Binary vs. Non-Binary Relationships

• Some relationships that appear to be non-binary may be better represented using binary relationships

  – E.g. A ternary relationship *parents*, relating a child to his/her father and mother, is best replaced by two binary relationships, *father* and *mother*
    • Using two binary relationships allows partial information (e.g. only mother being known)

  – But there are some relationships that are naturally non-binary
    • Example: *works_on*
Converting Non-Binary Relationships to Binary Form

- In general, any non-binary relationship can be represented using binary relationships by creating an artificial entity set.
  - Replace \( R \) between entity sets \( A \), \( B \) and \( C \) by an entity set \( E \), and three relationship sets:
    1. \( R_A \), relating \( E \) and \( A \)
    2. \( R_B \), relating \( E \) and \( B \)
    3. \( R_C \), relating \( E \) and \( C \)
  - Create a special identifying attribute for \( E \)
  - Add any attributes of \( R \) to \( E \)
  - For each relationship \((a_i, b_i, c_i)\) in \( R \), create
    1. a new entity \( e_i \) in the entity set \( E \)
    2. add \((e_i, a_i)\) to \( R_A \)
    3. add \((e_i, b_i)\) to \( R_B \)
    4. add \((e_i, c_i)\) to \( R_C \)
From E/R Diagrams to Relations

- Entity set $\rightarrow$ relation.
  - Attribute of an entity set $\rightarrow$ attribute of a relation.
- Relationship $\rightarrow$ relation whose attributes are
  - Attribute of the relationship itself.
  - Key attributes of the connected entity sets.
- Several special cases:
  - Weak entity sets.
  - Combining relations (especially for many-one relationships).
  - $\text{i}sa$ relationships and subclasses.
Schemas for Non-Weak Entity Sets

- For each entity set, create a relation with the same name and with the same set of attributes.

  Students(Name, Address)

  Professors(Name, Office, Age)

  Departments(Name)
Schemas for Weak Entity Sets

- For each weak entity set $W$, create a relation with the same name whose attributes are
  - Attributes of $W$ and
  - Key attributes of the other entity sets that help form the key for $W$.

Courses(Number, DepartmentName, CourseName, Classroom, Enrollment)
Schemas for Non-Supporting Relationships

- For each relationship, create a relation with the same name whose attributes are
  - Attributes of the relationship itself.
  - Key attributes of the connected entity sets (even if they are weak).
Schemas for Non-Supporting Relationships

- Take(StudentName, Address, Number, DepartmentName)
- Teach(ProfessorName, Office, Number, DepartmentName)
- Evaluation(StudentName, Address, ProfessorName, Office, Number, DepartmentName, Grade)
Roles in Relationships

If an entity set $E$ appears $k > 1$ times in a relationship $R$ (in different roles), the key attributes for $E$ appear $k$ times in the relation for $R$, appropriately renamed.

PreReq(RequirerNumber, RequirerDeptName, RequirementNumber, RequirementDeptName)
Combining Relations

- Consider many-one Teach relationship from Courses to Professors.
- Schemas are:
  
  Courses(Number, DepartmentName, CourseName, Classroom, Enrollment)
  Professors(Name, Office, Age)
  Teach(Number, DepartmentName, ProfessorName, Office)

- The key for Courses uniquely determines all attributes of Teach.
- We can combine the relations for Courses and Teach into a single relation whose attributes are:
  - All the attributes for Courses,
  - Any attributes of Teach, and
  - The key attributes of Professors.
Rules for Combining Relations

- We can combine into one relation $Q$
  - The relation for an entity set $E$ and
  - all many-to-one relationships $R_1, R_2, \ldots, R_k$ from $E$ to other entity sets $E_1, E_2, \ldots, E_k$, respectively.

- The attributes of $Q$ are
  - all the attributes of $E$,
  - any attributes of $R_1, R_2, \ldots, R_k$, and
  - the key attributes of $E_1, E_2, \ldots, E_k$. 
Supporting Relationships

- Schema for Departments is Departments(Name).
- Schema for Courses is Courses(Number, DepartmentName, CourseName, Classroom, Enrollment).
- What is the schema for Offer?
• Offer(Name, Number, DepartmentName).

• But Name and DepartmentName are identical, so the schema for Offer is Offer(Number, DepartmentName).

The schema for Offer is a subset of the schema for the weak entity set, so we can dispense with the relation for Offer.
End of E/R Diagrams