Outline

- Relational Algebra (6.1)
- E/R Model (7.2 - 7.4)
- E/R Diagrams (7.5)
- Reduction to Schema (7.6)
- Relational Database Design (7.7)
- Functional Dependencies (8.1 – 8.4)
- Normalization (8.5 – 8.7)
- Relational Query Languages
- SQL Basics
- Formal Semantics of SQL

E-R Diagrams

Rectangles represent entity sets.
Diamonds represent relationship sets.
Attributes listed inside entity rectangle
Underline indicates primary key attributes
Composite, Multivalued, and Derived

<table>
<thead>
<tr>
<th>instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>first_name</td>
</tr>
<tr>
<td>middle_initial</td>
</tr>
<tr>
<td>last_name</td>
</tr>
<tr>
<td>address</td>
</tr>
<tr>
<td>street</td>
</tr>
<tr>
<td>street_number</td>
</tr>
<tr>
<td>street_name</td>
</tr>
<tr>
<td>apt_number</td>
</tr>
<tr>
<td>city</td>
</tr>
<tr>
<td>state</td>
</tr>
<tr>
<td>zip</td>
</tr>
<tr>
<td>phone_number</td>
</tr>
<tr>
<td>date_of_birth</td>
</tr>
<tr>
<td>age ()</td>
</tr>
</tbody>
</table>

Relationship Sets with Attributes

<table>
<thead>
<tr>
<th>instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>salary</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>advisor</th>
</tr>
</thead>
</table>

| date |

<table>
<thead>
<tr>
<th>student</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>tot_cred</td>
</tr>
</tbody>
</table>

Roles

- Entity sets of a relationship need not be distinct
  - Each occurrence of an entity set plays a “role” in the relationship
- The labels “course_id” and “prereq_id” are called roles.

Cardinality Constraints

- We express cardinality constraints by drawing either a directed line (→), signifying “one,” or an undirected line (—), signifying “many,” between the relationship set and the entity set.
- One-to-one relationship:
  - A student is associated with at most one instructor via the relationship advisor
  - A student is associated with at most one department via stud_dept
One-to-One Relationship

- one-to-one relationship between an instructor and a student
  - an instructor is associated with at most one student via advisor
  - a student is associated with at most one instructor via advisor

One-to-Many Relationship

- one-to-many relationship between an instructor and a student
  - an instructor is associated with any number (including 0) of students via advisor
  - a student is associated with at most one instructor via advisor,
Many-to-One Relationships

- In a many-to-one relationship between an instructor and a student,
  - an instructor is associated with at most one student via advisor,
  - and a student is associated with several (including 0) instructors via advisor.

Many-to-Many Relationship

- An instructor is associated with any number (possibly 0) of students via advisor.
- A student is associated with any number (possibly 0) of instructors via advisor.
**Participation** of an Entity Set in a Relationship Set

- Total participation (indicated by **double line**): every entity in the entity set participates in at least one relationship in the relationship set
- E.g., participation of *section* in *sec_course* is total
  - every *section* must have an associated course
- Partial participation: some entities may not participate in any relationship in the relationship set
- Example: participation of *course* in *section* is partial

![Diagram of Course and Section entities with relationship sec_course]

**Alternative Notation for Cardinality Limits**

- Cardinality limits can also express participation constraints

![Diagram of Instructor, Advisor, and Student entities with relationship advisor]
What attributes are needed to represent a relationship completely and uniquely?

- Union of primary entities keys, and relationship attributes

- \{i.ID, s.ID, date\} describes relationship completely

Is \{i.ID, s.ID, date\} a candidate key for advisor?

- No. Attribute access-date can be removed from this set without losing key-ness
- In fact, union of primary keys of associated entities is always a superkey

Almost always.
Is \{i.ID, s.ID\} a candidate key?

- Depends...

If one-to-many relationship, \{s.ID\} is a candidate key
- Assumes an instructor can advise many students, but students each have at most one advisor.
Is \{i.ID, s.ID\} a candidate key?
  - Depends...

- If one-to-one relationship, either \{i.ID\} or \{s.ID\} sufficient
  - Since a given advisor can only advise a single student, she can only participate in one relationship
  - Same for student

**General rule for binary relationships**
- one-to-one: primary key of either entity set
- one-to-many: primary key of the entity set on the many side
- many-to-many: union of primary keys of the associate entity sets

**n-ary relationships**
- More complicated rules
Summary: Keys for Relationship Sets

- Primary keys of entity sets together form super key of a relationship set.
  - \((s\_id, i\_id)\) is the super key of advisor
  - **NOTE:** this means *an entity pair have at most one relationship in a relationship set.*
    - Example: if we wish to track multiple meeting dates between a student and her advisor, we cannot assume a relationship for each meeting.
    - Fix: use a multivalued attribute.
- Must consider the mapping cardinality of the relationship set when identifying candidate keys.
- Consider relationship set semantics in selecting the primary key if more than one candidate key.

Weak Entity Sets

- An entity set that does not have a primary key is referred to as a **weak entity set.**
- The existence of a weak entity set depends on the existence of an identifying entity set
  - It must relate to the identifying entity set via a one-to-many relationship set from the identifying to the weak entity set
  - Weak side of relationship set must be total
    - Identifying relationship depicted using a double diamond
- The **discriminator** (or partial key) of a weak entity set is the set of attributes that *help* distinguish among all the entities of a weak entity set.
- Primary key of a weak entity set is:
  - primary key of the strong entity set on which the weak entity set is existent-dependent,
  - plus the weak entity set’s discriminator.
We underline the discriminator of a weak entity set with a dashed line.

We put the identifying relationship of a weak entity in a double diamond.

Primary key for section – \((course\_id, sec\_id, semester, year)\)

Note: the primary key of the strong entity set is not explicitly stored with the weak entity set, since it is implicit in the identifying relationship.

If \(course\_id\) were explicitly stored, \(section\) could be made a strong entity, but then the relationship between \(section\) and \(course\) would be duplicated by an implicit relationship defined by the attribute \(course\_id\) common to \(course\) and \(section\)

Duplication is bad
Summary of E-R Notation

- **Entity set (E)**
- **Relationship set (R)**
- **Identifying relationship set for weak entity set**
- **Total participation of entity set in relationship**

**Attributes**:
- Simple (A1)
- Composite (A2)
- Multivalued (A3)
- Derived (A4)

**Primary key**

**Discriminating attribute of weak entity set**

Symbols Used in E-R Notation (Cont.)

- **Many-to-many relationship**
- **Many-to-one relationship**
- **One-to-one relationship**
- **Role indicator**
Suppose we have entity sets

- instructor, with attributes including dept_name
- department

and a relationship

- inst_dept relating instructor and department

Attribute dept_name in entity instructor is redundant since there is an explicit relationship inst_dept which relates instructors to departments

- The attribute replicates information present in the relationship, and should be removed from instructor

- BUT: redundant attributes sometimes get reintroduced when converting back to tables
Reduction to Relational Schemas

Outline

- Relational Algebra (6.1)
- E/R Model (7.2 - 7.4)
- E/R Diagrams (7.5)
- Reduction to Schema (7.6)
- Relational Database Design (7.7)
- Functional Dependencies (8.1 – 8.4)
- Normalization (8.5 – 8.7)
Reduction to Relation Schemas

- Entity sets and relationship sets can be expressed uniformly as relation schemas that represent the contents of the database.
- A database which conforms to an E-R diagram can be represented by a collection of schemas.
- For each entity set and relationship set there is a unique schema that is assigned the name of the corresponding entity set or relationship set.
- Each schema has a number of columns (generally corresponding to attributes), which have unique names.

Representing Entity Sets

- A strong entity set reduces to a schema with the same attributes student(ID, name, tot_cred)
- A weak entity set becomes a table that includes a foreign key for the primary key of the identifying strong entity set section (course_id, sec_id, sem, year)
Representing Relationship Sets

- A many-to-many relationship set is represented as a schema with attributes for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set.
- Example: schema for relationship set advisor:

\[ \text{advisor}(s\_id, i\_id) \]

Redundancy of Schemas

- Many-to-one and one-to-many relationship sets that are total on the many-side can be represented by adding an extra attribute(s) to the “many” side, containing the primary key of the “one” side
- Example:
  - get rid of inst_dept by:

\[ \text{instructor}(ID, \text{name}, \text{salary}) \rightarrow \text{instructor}(ID, \text{dept\_name}, \text{name}, \text{salary}) \]
Redundancy of Schemas (Cont.)

- For one-to-one relationship sets, either side can be chosen to act as the “many” side
  - That is, extra attribute can be added to either of the tables corresponding to the two entity sets

  ![Diagram](image)

  instructor(ID, dept_name, name, salary)  
  
or  
  department(dept_name, ID, building, budget)

Redundancy of Schemas (Cont.)

- If participation is partial on the “many” side, replacing a relationship schema by an extra attribute in the schema corresponding to the “many” side could result in null values (generally avoided)
  - i.e. the approach in the previous slides does not work
  - need to represent relationship as a separate table

- Relationship set linking a weak entity set to its identifying strong entity set is redundant.
  - Example: The section schema already contains the attributes that would appear in the sec_course schema
  - Unless otherwise instructed, assume we wish to avoid NULLs when converting to relations, i.e. remove redundant relationship schemas only when total participation on side where adding attribute.
**Composite Attributes**

- Composite attributes flattened out
  - Example: given entity set `instructor`
    - with composite attribute `name`
    - with component attributes `first_name` and `last_name`
    - replace with `name_first_name` and `name_last_name`
    - Prefix omitted if there is no ambiguity
  - Ignoring multivalued attributes, extended instructor schema is
    - `instructor(ID, first_name, middle_initial, last_name, street_number, street_name, apt_number, city, state, zip_code, date_of_birth)`

---

**Multivalued Attributes**

- Multivalued attribute `M` of entity `E` represented by a separate schema `EM`
  - Schema `EM` includes `E`'s primary key and attribute corresponding to `M`
  - Example: Multivalued attribute `phone_number` of `instructor`:
    - `inst_phone= (ID, phone_number)`

  - Each value of the multivalued attribute maps to separate tuple of `EM`:
    - `instructor` entity with primary key `22222` and numbers 456-7890 and 123-4567 maps to:
      - `(22222, 456-7890)`
      - `(22222, 123-4567)`
Reduction to Relational Schemas

Outline

- Relational Algebra (6.1)
- E/R Model (7.2 - 7.4)
- E/R Diagrams (7.5)
- Reduction to Schema (7.6)
- Relational Database Design (7.7)
- Functional Dependencies (8.1 – 8.4)
- Normalization (8.5 – 8.7)
ER Diagram to Relational Schema

- Schema per entity set
  - expand composite attributes
  - new schema for multi-valued
  - drop derived attributes for now

- Schema per relationship set

lots of foreign key dependences (weak, relationships...)

department(dept_name, building, budget)
instructor(id, name, salary)
course(course_id, title, credits)
section
student(id, name, tot_cred)
classroom(building, room_number, capacity)
time_slot(time_slot_id, ((day, start_time, end_time)))
inst_dept
stud_dept
teaches
takes
advisor(i_id, s_id)
course_dept(course_id, dept_name)
sec_time_slot(course_id, sec_id, semester, year, time_slot_id)
sec_course( mess )
prereq(course_id, prereq_id)
sec_class(course_id, sec_id, semester, year, building, room_number)

department(dept_name, building, budget)
instructor(id, name, salary)
course(course_id, title, credits)
section
student(id, name, tot_cred)
classroom(building, room_number, capacity)
time_slot(time_slot_id, day, start_time, end_time)
inst_dept(id, dept_name)
stud_dept(id, dept_name)
teaches(id, sec_id, semester, course_id, year)
takes(id, course_id, sec_id, semester, year, grade)
advisor(i_id, s_id)
course_dept(course_id, dept_name)
sec_time_slot(course_id, sec_id, semester, year, time_slot_id)
sec_course( mess )
prereq(course_id, prereq_id)
sec_class(course_id, sec_id, semester, year, building, room_number)

lots of foreign key dependences (weak, relationships...)
ER Diagram to Relational Schema

- Schema per entity set
  - expand composite attributes
  - new schema for multi-valued
  - drop derived attributes for now
- Schema per relationship set

lots of foreign key dependences (weak, relationships..)

department(dept_name, building, budget)
instructor(ID, name, salary)
course(course_id, title, credits)
section(sec_id, course_id, semester, year)
student(ID, name, tot_cred)
classroom(building, room_number, capacity)
time_slot(time_slot_id, day, start_time, end_time)
inst_dept(ID, dept_name)
stud_dept(ID, dept_name)
teaches(ID, sec_id, semester, course_id, year)
takes(ID, course_id, sec_id, semester, year, grade)
advisor(i_id, s_id)
course_dept(course_id, dept_name)
sec_time_slot(course_id, sec_id, semester, year, time_slot_id)
sec_course(course_id, sec_id, semester, year)
prereq(course_id, prereq_id)
sec_class(course_id, sec_id, semester, year, building, room_number)

ds家伙的外键依赖关系 (脆弱, 简单的...)

department(dept_name, building, budget)
instructor(ID, name, salary)
course(course_id, title, credits)
section(sec_id, course_id, semester, year)
student(ID, name, tot_cred)
classroom(building, room_number, capacity)
time_slot(time_slot_id, day, start_time, end_time)
inst_dept(ID, dept_name)
stud_dept(ID, dept_name)
teaches(ID, sec_id, semester, course_id, year)
takes(ID, course_id, sec_id, semester, year, grade)
advisor(i_id, s_id)
course_dept(course_id, dept_name)
sec_time_slot(course_id, sec_id, semester, year, time_slot_id)
sec_course(course_id, sec_id, semester, year)
prereq(course_id, prereq_id)
sec_class(course_id, sec_id, semester, year, building, room_number)

ds家伙的外键依赖关系 (脆弱, 简单的...)

ER Diagram to Relational Schema

- Schema per entity set
  - expand composite attributes
  - new schema for multi-valued
  - drop derived attributes for now
- Schema per relationship set

lots of foreign key dependences (weak, relationships..)

department(dept_name, building, budget)
instructor(ID, dept_name, name, salary)
course(course_id, title, credits)
section(sec_id, course_id, semester, year)
student(ID, name, tot_cred)
classroom(building, room_number, capacity)
time_slot(time_slot_id, day, start_time, end_time)

inst_dept(ID, dept_name)
stud_dept(ID, dept_name)
teaches(ID, sec_id, semester, course_id, year)
takes(ID, course_id, sec_id, semester, year, grade)
advisor(i_id, s_id)
course_dept(course_id, dept_name)
sec_time_slot(course_id, sec_id, semester, year, time_slot_id)
sec_course(course_id, sec_id, semester, year)
prereq(course_id, prereq_id)
sec_class(course_id, sec_id, semester, year, building, room_number)

department(dept_name, building, budget)
instructor(ID, dept_name, name, salary)
course(course_id, title, credits)
section(sec_id, course_id, semester, year)
student(ID, dept_name, name, tot_cred)
classroom(building, room_number, capacity)
time_slot(time_slot_id, day, start_time, end_time)

inst_dept(ID, dept_name)
stud_dept(ID, dept_name)
teaches(ID, sec_id, semester, course_id, year)
takes(ID, course_id, sec_id, semester, year, grade)
advisor(i_id, s_id)
course_dept(course_id, dept_name)
sec_time_slot(course_id, sec_id, semester, year, time_slot_id)
sec_course(course_id, sec_id, semester, year)
prereq(course_id, prereq_id)
sec_class(course_id, sec_id, semester, year, building, room_number)

lots of foreign key dependences (weak, relationships..)
ER Diagram to Relational Schema

- Schema per entity set
  - expand composite attributes
  - new schema for multi-valued
  - drop derived attributes for now

- Schema per relationship set

lots of foreign key dependences (weak, relationships)

department(dept_name, building, budget)
instructor(i_id, dept_name, name, salary)
course(id, course_id, title, credits, dept_name)
section(sec_id, course_id, semester, year)
student(s_id, dept_name, name, tot_cred)
classroom(building, room_number, capacity)
time_slot(time_slot_id, day, start_time, end_time)

inst_dept(id, dept_name)
stud_dept(id, dept_name)
teaches(id, sec_id, semester, course_id, year)
takes(id, course_id, sec_id, semester, year, grade)
advisor(i_id, s_id)
course_dept(course_id, dept_name)
sec_time_slot(course_id, sec_id, semester, year)
time_slot_id)
sec_course(course_id, sec_id, semester, year)
prereq(course_id, prereq_id)
sec_class(course_id, sec_id, semester, year, building, room_number)

lots of foreign key dependences (weak, relationships)
ER Diagram to Relational Schema

- Schema per entity set
  - expand composite attributes
  - new schema for multi-valued
  - drop derived attributes for now
- Schema per relationship set

  ![ER Diagram]

- department(dept_name, building, budget)
- instructor(iD, dept_name, name, salary)
- course(course_id, title, credits, dept_name)
- section(sec_id, course_id, semester, year, building, room_number, time_slot_id)
- student(iD, dept_name, name, tot_cred)
- classroom(building, room_number, capacity)
- time_slot(time_slot_id, day, start_time, end_time)
- inst_dept(iD, dept_name)
- stud_dept(iD, dept_name)
- teaches(iD, sec_id, semester, course_id, year)
- takes(iD, course_id, sec_id, semester, year, grade)
- advisor(iD, s_iD)
- course_dept(course_id, dept_name)
- sec_time_slot(course_id, sec_id, semester, year, time_slot_id)
- sec_course(course_id, sec_id, semester, year)
- prereq(course_id, prereq_id)
- sec_class(building, room_number, capacity, building, room_numbers)

Done. lots of foreign key dependences (weak, relationships...)

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: proj_group, with several project members
Design Issues

- **Binary versus n-ary relationship sets**
  Although it is possible to replace any nonbinary (n-ary, for \( n > 2 \)) relationship set by a number of distinct binary relationship sets, a n-ary relationship set shows more clearly that several entities participate in a single relationship.

- **Placement of relationship attributes**
  e.g., attribute *date* as attribute of *advisor* or as attribute of *student*

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 \( R_A, R_B, R_C \), relating \( E \) with \( A, B, \) 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 a new entity \( e_i \) in the entity set \( E \)
    - add \((e_i, a_i)\) to \( R_A \), etc.

(a) (b)