# COMP353 Databases

Relational Data Model

# Conceptual Database Design

## Relational Data Model:

Introduction

## The Relational Data Model

- **Relational database**
  - A set of **relations**
- **Relation**
  - A two-dimensional table in which data is arranged

## Example: Relation

Attribute **Names**

| Title | Year | Length | FilmType |
|---|---|---|---|
| Star Wars | 1997 | 124 | color |
| Mighty Ducks | 1991 | 104 | color |
| Wayne's World | 1992 | 95 | color |
| . . . | . . . | . . . | . . . |

Each row is a tuple

Components of the tuple

Attributes (type) are atomic (1NF)

## Relational Data Model

- Relation schema (or structure): $R_i=\{A_1,\ldots,A_m\}$
  - Relation name + a set of attribute names (+ attribute types)
- Relation instance:
  - The set of "current" tuples
- Database schema:
  - A set of relation schemas $D=\{R_1,\ldots,R_n\}$
- Database instance:
  - A collection of relation instances -- one for each relation in the database schema

## Relational Query Languages

- A major strength of the relational model is that it supports a powerful, high-level programming language – the Structured Query Language (SQL)

# Logical Database Design

## From E/R to Relational Model

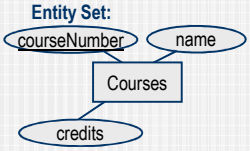
## Converting E/R to Relational Model

- Input:
  - An E/R diagram
- Output:
  - A relational database schema -- a collection of relations

## Converting Entity Sets to Relations

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

**Entity Set:**




**Relation Instance:**

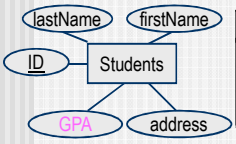
| courseNumber | name | credits |
|---|---|---|
| Comp248 | C++ Prog. | 3 |
| Comp352 | Data Structures | 4 |
| Comp353 | Databases | 4 |

**Relation Schema:**

In theory, **Courses** = {courseNumber, name, credits}
In practice, **Courses**(courseNumber, name, credits)

## Converting Entity Sets to Relations

**Entity Set:**




**Relation Instance:**

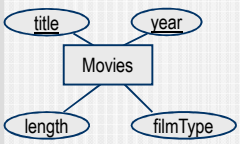
| ID | firstName | lastName | GPA | address |
|---|---|---|---|---|
| 111 | Joe | Smith | 4.0 | 45 Pine av. |
| 222 | Sue | Brown | 3.1 | 71 Main St. |
| 333 | Ann | John | 3.7 | 39 Bay St. |

**Relation Schema:**

**Students**(ID, firstName, lastName, GPA, address)

## Converting Entity Sets to Relations

**Entity Set:**




**Relation Instance:**

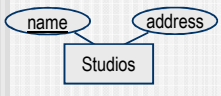
| title | year | length | filmType |
|---|---|---|---|
| Star Wars | 1997 | 124 | color |
| Mighty Ducks | 1991 | 104 | color |
| Wayne's World | 1992 | 95 | b&w |

**Relation Schema:**

**Movies**(title, year, length, filmType)

## Converting Entity Sets to Relations

**Entity Set:**




**Relation Instance:**

| name | address |
|---|---|
| Fox | Hollywood |
| Disney | Hollywood |
| Paramount | Hollywood |

**Relation Schema:**

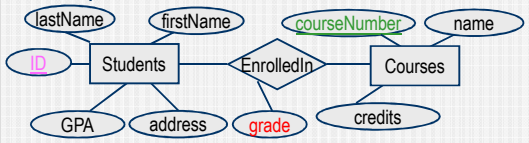
**Studios**(name, address)

## Converting Relationships to Tables

- For each relationship set **R**, create a relation (table) with the same name **R**
- The set of attributes of this relation includes:
  - Key attribute(s) of each entity set involved in the relationship
  - The "explicit" attributes which **R** may have

## From Relationships to Tables

**Relationship Set:**




**Relation Schema:**

**EnrolledIn** (ID, courseNumber, grade)

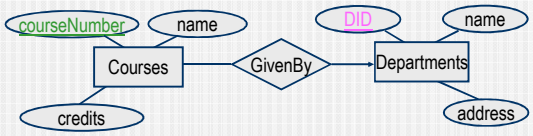
**What is the primary key of this relation?**

**Relation Instance:**

| ID | courseNumber | grade |
|---|---|---|
| 123 | Comp248 | A- |
| 456 | Comp248 | B |
| 123 | Comp353 | A+ |

## From Relationships to Tables

**Relationship Set:**




**Relation Schema:**

**GivenBy**(courseNumber, DID)

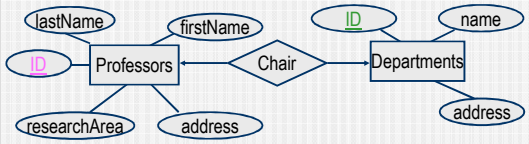
**What is the primary key of this relation?**

**Relation Instance:**

| courseNumber | DID |
|---|---|
| Comp248 | 1 |
| Comp352 | 1 |
| Math207 | 9 |

## From Relationships to Tables

**Relationship Set:**




**Relation Schema:**

**Chair**(PID, DID)

**What is the primary key of this relation?**

**Relation Instance:**

| PID | DID |
|---|---|
| 234 | 1 |
| 451 | 2 |
| 778 | 9 |

## Identifying Key of Relationship R

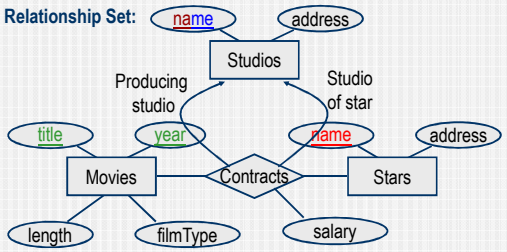
- We are already familiar with the concept of key
- If **R** is a **binary** relationship between entity sets **E1** and **E2**, then the multiplicity of this relationship determines the key of **R**
  - If **R** is M-N, then the keys of **E1** and **E2** together are "part of" the key of **R**
  - If **R** is M-1 from **E1** to **E2**, then the key of **E1** is part of the key of **R**
  - If **R** is 1-1, then either **E1** or **E2** (but not both) is part of the key of **R**
- Do the above rules regarding the formation of keys apply to:
  - **Multi-way relationships?**
- How to determine keys for:
  - **Week entity sets?**
  - **Entity sets *and* relationship *sets in isa* hierarchies?**

## Converting Relationships to Tables

- We should **rename** the attributes in the relations created when:
  - An entity set is involved in a relationship more than once
  - The **same attribute name** appears in the keys of different entity sets involved in the relationship (e.g., ID in previous example)
  - This is to avoid **ambiguity** in the schema and to be more clear in meanings

## Relationship Sets to Relations

**Relationship Set:**




**Relation Schema:**

**Contracts**(starName, title, year, studioOfStar, producingStudio, salary )

**What is the primary key for Contracts?**

## Weak Entity Sets to Relations

- The relation/table **W** for the weak entity set $W$, must include all the attributes of $W$ as well as the key attributes of the strong entity sets to which $W$ is associated.
- Any relationship R to which the weak entity set $W$ contributes, must include **all** the key attributes of $W$, i.e., the key attributes of *every* entity set that contributes to $W$'s key
- The weak relationships, from the weak entity set $W$ to other entity sets that provide the key for $W$, need not be converted into a separate table, i.e., double diamonds connecting a weak entity set need not become a separate table.

## Weak Entity Sets to Relations

**Weak Entity Set:**

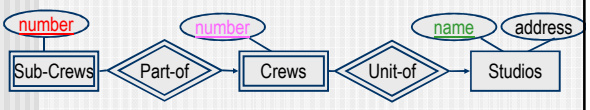

**Relation Schema:**

**Crews** (number, name)
**Unit-of** (number, studioName, name)
**Studios** (name, address)

**Do we need to keep the relation Unit-of?**

## Weak Entity Sets to Relations

**Weak Entity Set:**

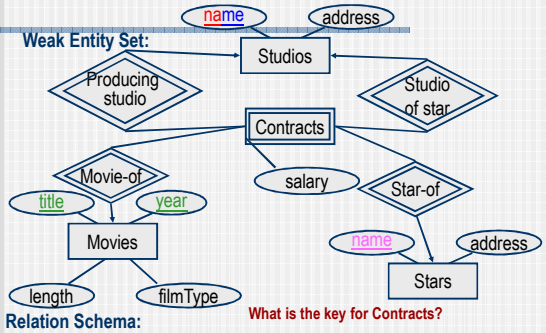



**Relation Schemas:**

**Sub-Crews** (number,crewNumber, name)
**Studios** (name, address)

## Weak Entity Sets to Relations

**Weak Entity Set:**




**Relation Schema:**

**Contracts**(starName, title, year, studioOfStar, producingStudio,salary)

**What is the key for Contracts?**

## Converting *isa*-Hierarchies to Relations

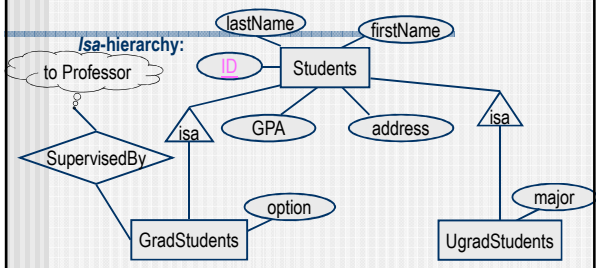
There are 3 approaches:

- Straight-E/R style method
  - In the E/R model, an entity (object) can be represented by entities that may belong to several entity sets, which are connected and related via ***isa*** hierarchies
  - The "connected" entities together represent the object and also determine the object's properties (e.g., attributes and relationships)
- The object-oriented method
- The nulls method

## Converting *isa*-Hierarchy to Relations

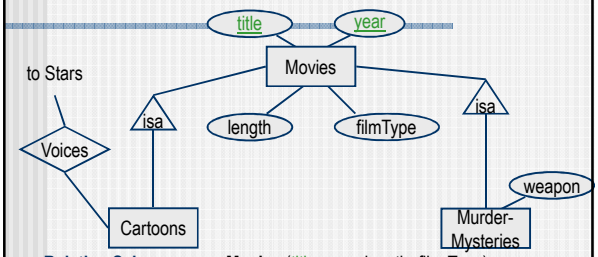
- For each entity set **E**, create a relation (table) **e**, and give it attribute(s) **A**, whenever:
  - **A** belongs to **E**
  - **A** is the key attribute of the parent(s) relation
- No relation is created for the ***isa***-relationship

## *Isa*-Hierarchy to Relations




**Relation Schemas:**
**Students** (ID, lastName, firstName, GPA, address )
**GradStudents** (ID, option )
**UgradStudents** (ID, major )
**SupervisedBy** (StudentID, professorID)

## *Isa*-Hierarchy to Relations



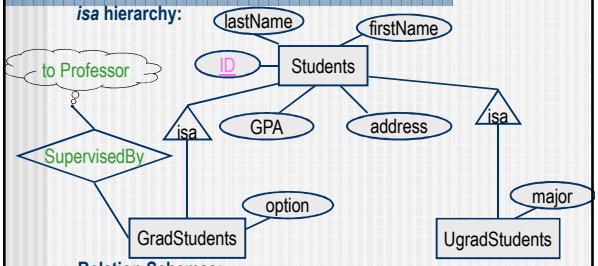

**Relation Schemas:**
**Movies** (title, year, length, film Type)
**Cartoons** (title, year) ← Do we really need this?
**Murder-Mysteries** (title, year, weapon)
**Voices** (starName, title, year)

What about Cartoon-Murder-Mysteries?

## The NULL Values Approach

- If we are **allowed** to use **NULL** as a value in tuples, we can handle a hierarchy of entity sets (classes) with a single relation
  - This relation has all the attributes belonging to any entity set (class) of the hierarchy.
  - An entity/object is represented by a single tuple that has NULL in each attribute that is not defined for that entity/object.

## Converting isa Hierarchy to Relations: The Null Approach




**Relation Schemas:**
**SupervisedBy** (ID, professorID)
**Student** (ID, lastName, firstName, GPA, address, option, major)

## NULL Approach

- The null approach: **supports efficient query processing but is inefficient in space utilization. Why?**
  - Answering queries: the nulls approach allows us to find, *in a **single** relation **R***, every tuple/object from any set involved in the hierarchy
  - Allows us to find **all** the information about **an entity/object** in a single tuple in **R**
  - The down side is its *space utilization which is too costly for having* repeated and redundant information:
  - **Note:** Nulls are not allowed in the relational model theory, but practically, it is supported by commercial DBMS

## A quick test!

- Suppose R is a M-1 relationship from entity set E1={a1,a2} to E2={b1,b2}. Which of the following is NOT a ***valid instance*** of R?
  - R = {(a1, b1), (a1, b2)}.
  - R = {(a1, b1)}.
  - R = {(a2, b1)}.
  - R = {(a1, b1), (a2, b1)}.
  - R = { }