ICS 321 Fall 2009
The Relational Model (ii)

Asst. Prof. Lipyeow Lim
Information and Computer Science Department
University of Hawaii at Manoa
Logical DB Design: ER to Relational

- Entity sets to tables:

```sql
CREATE TABLE Employees
(ssn CHAR(11),
name CHAR(20),
lot INTEGER,
PRIMARY KEY (ssn))
```
Relationship Sets to Tables

- In translating a relationship set to a relation, attributes of the relation must include:
  - Keys for each participating entity set (as foreign keys).
    - This set of attributes forms a superkey for the relation.
  - All descriptive attributes.

```sql
CREATE TABLE Works_In(
    ssn CHAR(11),
    did INTEGER,
    since DATE,
    PRIMARY KEY (ssn, did),
    FOREIGN KEY (ssn)
        REFERENCES Employees,
    FOREIGN KEY (did)
        REFERENCES Departments)
```
Review: Key Constraints

- Each dept has at most one manager, according to the key constraint on Manages.
Translating ER Diagrams with Key Constraints

• Map relationship to a table:
  – Note that `did` is the key now!
  – Separate tables for Employees and Departments.
• Since each department has a unique manager, we could instead combine Manages and Departments.

```sql
CREATE TABLE Manages(
  ssn CHAR(11),
  did INTEGER,
  since DATE,
  PRIMARY KEY (did),
  FOREIGN KEY (ssn) REFERENCES Employees,
  FOREIGN KEY (did) REFERENCES Departments)
```

```sql
CREATE TABLE Dept_Mgr(
  did INTEGER,
  dname CHAR(20),
  budget REAL,
  ssn CHAR(11),
  since DATE,
  PRIMARY KEY (did),
  FOREIGN KEY (ssn) REFERENCES Employees
)```
Review: Participation Constraints

• Does every department have a manager?
  – If so, this is a **participation constraint**: the participation of Departments in Manages is said to be **total** (vs. **partial**).
  • Every *did* value in Departments table must appear in a row of the Manages table (with a non-null *ssn* value!)
Participation Constraints in SQL

• We can capture participation constraints involving one entity set in a binary relationship, but little else (without resorting to CHECK constraints).

```sql
CREATE TABLE Dept_Mgr(
    did INTEGER,
    dname CHAR(20),
    budget REAL,
    ssn CHAR(11) NOT NULL,
    since DATE,
    PRIMARY KEY (did),
    FOREIGN KEY (ssn) REFERENCES Employees,
    ON DELETE NO ACTION)
```
Review: Weak Entities

- A weak entity can be identified uniquely only by considering the primary key of another (owner) entity.
  - Owner entity set and weak entity set must participate in a one-to-many relationship set (1 owner, many weak entities).
  - Weak entity set must have total participation in this identifying relationship set.

![Entity-Relationship Diagram](link-to-diagram)
Translating Weak Entity Sets

• Weak entity set and identifying relationship set are translated into a single table.
  – When the owner entity is deleted, all owned weak entities must also be deleted.

CREATE TABLE Dep_Policy (  
  pname CHAR(20),  
  age INTEGER,  
  cost REAL,  
  ssn CHAR(11) NOT NULL,  
  PRIMARY KEY (pname, ssn),  
  FOREIGN KEY (ssn) REFERENCES Employees,  
  ON DELETE CASCADE)
Review: ISA Hierarchies

- As in C++, or other PLs, attributes are inherited.
- If we declare A ISA B, every A entity is also considered to be a B entity.

- **Overlap constraints**: Can Joe be an Hourly_Emps as well as a Contract_Emps entity? *(Allowed/disallowed)*
- **Covering constraints**: Does every Employees entity also have to be an Hourly_Emps or a Contract_Emps entity? *(Yes/no)*
Translating ISA Hierarchies to Relations

• **General approach:**
  – 3 relations: Employees, Hourly_Emps and Contract_Emps.
    • *Hourly_Emps:* Every employee is recorded in Employees. For hourly emps, extra info recorded in Hourly_Emps (*hourly_wages, hours_worked, ssn*); must delete Hourly_Emps tuple if referenced Employees tuple is deleted).
    • Queries involving all employees easy, those involving just Hourly_Emps require a join to get some attributes.

• **Alternative:** Just Hourly_Emps and Contract_Emps.
  – *Hourly_Emps:* *ssn*, name, lot, *hourly_wages, hours_worked*.
  – Each employee must be in one of these two subclasses.
Review: Binary vs. Ternary Relationships

• What are the additional constraints in the 2nd diagram?
The key constraints allow us to combine Purchaser with Policies and Beneficiary with Dependents.

Participation constraints lead to NOT NULL constraints.

What if Policies is a weak entity set?

```sql
CREATE TABLE Policies (  
policyid INTEGER,  
cost REAL,  
ssn CHAR(11) NOT NULL,  
PRIMARY KEY (policyid).  
FOREIGN KEY (ssn) REFERENCES Employees,  
ON DELETE CASCADE)
```

```sql
CREATE TABLE Dependents (  
pname CHAR(20),  
age INTEGER,  
policyid INTEGER,  
PRIMARY KEY (pname, policyid).  
FOREIGN KEY (policyid) REFERENCES Policies,  
ON DELETE CASCADE)
```
Views

- A **view** is just a relation, but we store a *definition*, rather than a set of tuples.

  ```sql
  CREATE VIEW YoungActiveStudents (name, grade) 
  AS SELECT S.name, E.grade 
  FROM Students S, Enrolled E 
  WHERE S.sid = E.sid and S.age<21
  ```

- Views can be dropped using the **DROP VIEW** command.
  - How to handle **DROP TABLE** if there’s a view on the table?
    - **DROP TABLE** command has options to let the user specify this.
Views and Security

- Views can be used to present necessary information (or a summary), while hiding details in underlying relation(s).
  - Given YoungStudents, but not Students or Enrolled, we can find students s who have are enrolled, but not the cid’s of the courses they are enrolled in.
Relational Model: Summary

• A tabular representation of data.
• Simple and intuitive, currently the most widely used.
• Integrity constraints can be specified by the DBA, based on application semantics. DBMS checks for violations.
  – Two important ICs: primary and foreign keys
  – In addition, we always have domain constraints.
• Powerful and natural query languages exist.
• Rules to translate ER to relational model
Course Project (30%)

- Form a group of 3. Consider
  - Schedule of group members
  - Interests
  - Using Laulima “Discussion List”->”Student Lounge” to find groups/members

- Propose a project and post in Laulima under “Class Discussion” with “Project:” as a prefix in the title by Oct 6:
  - Title
  - Group name and members
    - Assign one person as the group representative
  - Short description of project goal
  - A brief timeline with milestones
    - Ideally you should have an idea who is doing what

- Do the project
- Prepare a short 10 min presentation and/or demo
- Present and demo your project in class on Dec 3 & 8.
Proposing a Project

• A good project would probably
  – Use at least one DBMS software
  – Involve substantial programming in SQL and another language (Java, C/C++, Ruby, PHP, PERL, ...)
  – Answer an interesting question or give insight into some specific issues

• Start by formulating the project goal or “question”, e.g.,
  – System building projects: how do we manage the data in application/scenario X? X should be interesting.
  – Researchy-type projects: what is the best way to do X?

• Evaluating “alternatives” should be an important component of the project.

• Think about how to quantify and measure your findings
Project Grading

• Originality
• Complexity
• Breadth & depth of the analysis/evaluation
• Presentation/demo