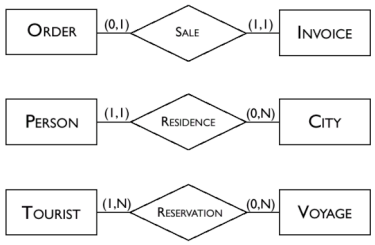
The Entity-Relationship Model

Part II

ER Cardinality Examples

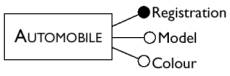


Textbook Notation

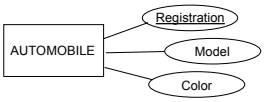
- …for (1,1) cardinalities (*key constraints*)

Examples of Keys

(internal) single-attribute key (unary key)



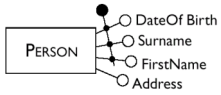
Slides Part 1 (UML inspired) notation

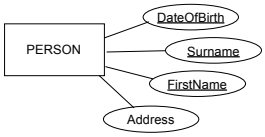


Text notation

Examples of Keys

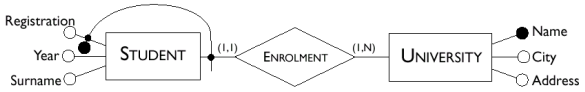
(Internal) multi-attribute (n-ary) key

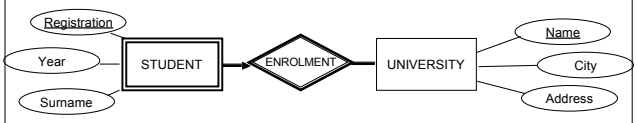




Examples of Keys

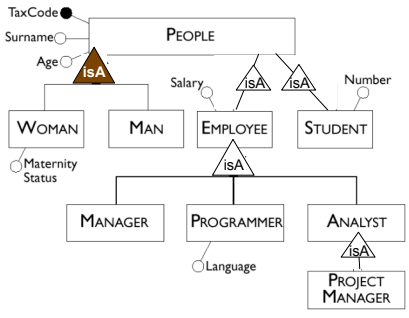
foreign, multi-attribute key (aka weak entity set)





- Arrow on line indicates *max cardinality* of 1 (key)
- Double (or thick) line indicates *min cardinality of 1*
  - aka *participation* constraint

An Entity Hierarchy



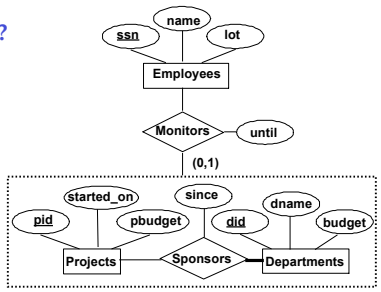
Aggregation

- Used when we have to model a relationship involving (entity sets and) and a *relationship set*.
- Aggregation allows us to treat a relationship set as an entity set for purposes of participation in other relationships.

An Example

Aggregation vs. ternary relationship?

- Monitors is a distinct relationship, with a descriptive attribute.
- Also, can say that each sponsorship is monitored by at most one employee.



Conceptual Design Using the ER Model

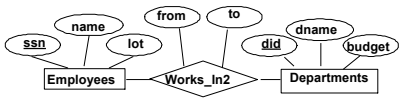
- Design choices:
  - ✓ Should a concept be modeled as an
    - ✓ entity or an attribute?
  - ✓ Should a concept be modeled as an
    - ✓ entity or a relationship?
  - ✓ Identifying relationships:
    - ✓ Binary or ternary? Aggregation?
- Note constraints of the ER Model:
  - ✓ A lot of data semantics can (and should) be captured.
  - ✓ But some constraints cannot be captured in ER diagrams.
    - ✓ We'll refine things in our logical (relational) design

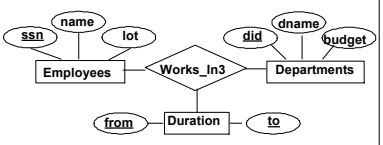
Entity vs. Attribute

- Should *address* be an attribute of Employees or an entity (related to Employees)?
- Depends upon how we want to use address information, and the semantics of the data:
  - ✓ If we have several addresses per employee, *address* must be an entity (since attributes cannot be set-valued).
  - ✓ If the structure (city, street, etc.) is important, *address* must be modeled as an entity (since attribute values are atomic).

Entity vs. Attribute (Cont.)

- Works_In2 does not allow an employee to work in a department for two or more periods.
- Similar to the problem of wanting to record several addresses for an employee: we want to record *several values of the descriptive attributes for each instance of this relationship.*



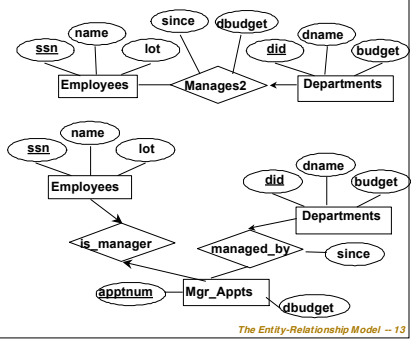


Entity vs. Relationship

OK as long as a manager gets a separate discretionary budget (*dbudget*) for each dept.

What if manager's *dbudget* covers *all* managed depts?

(can repeat value, but such redundancy is problematic)



Now you try it

Courses database:

- Courses, Students, Professors
- Courses have ids, titles, credits. The id is unique.
- Courses have multiple sections that have time, a room and exactly one teacher
- Professors have a unique name
- Students take courses and receive a grade
- Students may repeat a course
- Must track students' course schedules and transcripts including grades, semester taken, etc.
- Must track which classes a professor has taught
- Database should work over multiple semesters

Summary of Conceptual Design

- ***Conceptual design*** **follows** ***requirements analysis*****,**
  - ✓Yields a high-level description of data to be stored
- **ER model popular for conceptual design**
  - ✓Constructs are expressive, close to the way people think about their applications.
  - ✓Note: There are many variations on ER model
    - ✓Both graphically and conceptually
- **Basic constructs:** ***entities, relationships*****, and** ***attributes*** **(of entities and relationships).**
- **Some additional constructs:** ***weak entities, ISA hierarchies*****, and** ***aggregation*****.**

Summary of ER (Cont.)

- **Several kinds of integrity constraints:**
  - ✓*key constraints (max cardinality 1)*
  - ✓*participation constraints (min cardinality 1)*
  - ✓*overlap/covering* for ISA hierarchies.
- **Some** ***foreign key constraints*** **are also implicit in the definition of a relationship set.**
- **Many other constraints (notably,** ***functional dependencies*****) cannot be expressed.**
- **Constraints play an important role in determining the best database design for an enterprise.**

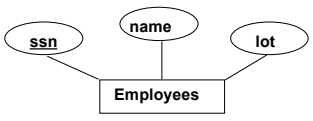
Summary of ER (Cont.)

- **ER design is *subjective*. There are often many ways to model a given scenario!**
- **Analyzing alternatives can be tricky, especially for a large enterprise. Common choices include:**
  - ✓ Entity vs. attribute, entity vs. relationship, binary or n-ary relationship, whether or not to use ISA hierarchies, aggregation.
- **Ensuring good database design: resulting relational schema should be analyzed and refined further.**
  - ✓ **Check for redundancy (see upcoming lectures)**

ER to Relational Mapping

Logical DB Design: ER to Relational

- Entity sets to tables.



ssn	name	lot
123-22-3666	Attishoo	48
231-31-5368	Smiley	22
131-24-3650	Smethurst	35

```
CREATE TABLE Employees
    (ssn CHAR(11),
    name CHAR(20),
    lot INTEGER,
    PRIMARY KEY (ssn))
```


Relationship Sets to Tables

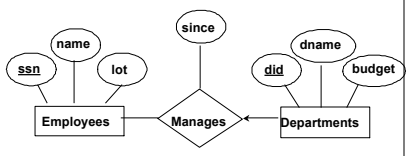
- In translating a many-to-many relationship set to a relation, attributes of the relation must include:
  - ✓ Keys for each participating entity set (as foreign keys).
    - ✓ This set of attributes forms a *superkey* for the relation.
  - ✓ All descriptive attributes.

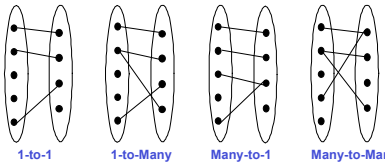
```
CREATE TABLE Works_In(
  ssn CHAR(1),
  did INTEGER,
  since DATE,
  PRIMARY KEY (ssn, did),
  FOREIGN KEY (ssn)
     REFERENCES Employees,
  FOREIGN KEY (did)
     REFERENCES Departments)
```

ssn	did	since
123-22-3666	51	1/1/91
123-22-3666	56	3/3/93
231-31-5368	51	2/2/92

Review: Key Constraints

- Each dept has *at most one* manager, according to the *key constraint* on Manages.





Alternative notation: (0,N) left, (0, 1) right

Translation to relational model?

Translating ER Diagrams with Key Constraints

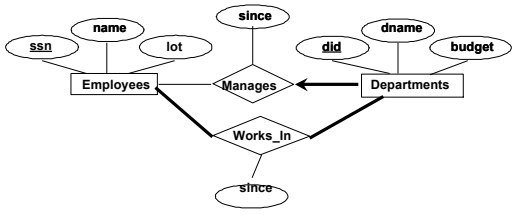
- Map relationship set to a table:
  - ✓ Note that did is the key now!
  - ✓ Separate tables for Employees and Departments.
- Since each department has a unique manager, we could instead combine Manages and Departments.

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

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

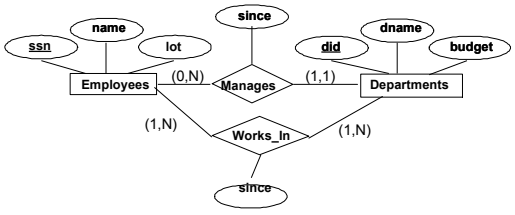

Review: Participation Constraints

- 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!)



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!)



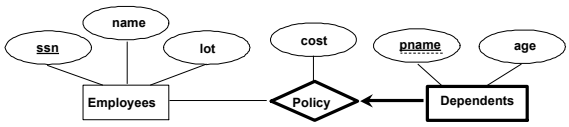
Participation Constraints in SQL

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

```
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,
)
```


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.



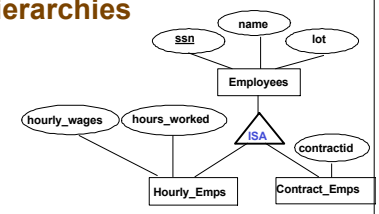
Translating Weak Entity Sets

- 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

- ❖ Attributes are inherited
  - ❖From superclass



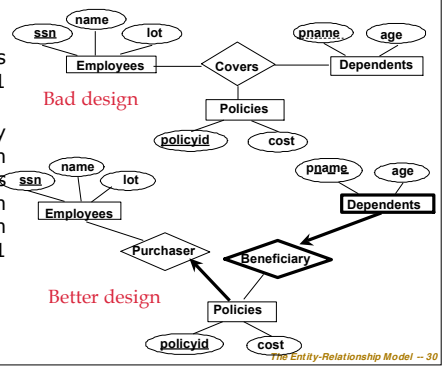
- *Overlap constraints*: Can Joe be an Hourly_Emps as well as a Contract_Emps entity? *(Allowed/disallowed)*
- *Covering constraints*: Does every Employees entity also have to be an Hourly_Emps or a Contract_Emps entity? *(Yes/no)*

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 Rel'nships

- If each policy is owned by just 1 employee:
  - ✓Key constraint on Policies would mean policy can only cover 1 dependent!



Binary vs. Ternary Relationships (Contd.)

- The key constraints allow us to combine Purchaser with Policies and Beneficiary with Dependents.
- Participation constraints lead to NOT NULL constraints.

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

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


ER Model Summary

- Usually easier to understand than Relational
- Expresses relationships clearly
- Rules to convert ER-diagrams to Relational Schema
- Some systems use ER-model for schema design
- Some people use ER-model as step before creating relational tables