Data Modeling and the Entity-Relationship Model

Spring 2018

School of Computer Science
University of Waterloo

Databases CS348

Outline

Overview of E-R Model

Used for (and designed for) database (conceptual schema) design

$\Rightarrow$ Proposed by Peter Chen in 1976

World/enterprise described in terms of

- entities
- attributes
- relationships

Visualization: **E-R diagram**

N.B. Many variant notations are in common use

Basic E-R Modeling

DB = (d, =, R1, R2)
eg 1.
Student/1
Flight/1
Burglary/1
{(s) | Student(s)}
{(e) | Student(e) $\wedge$ Fight(e)} = Ψ
if DB |= $\forall$e (Student(e)-> ¬Flight(e))

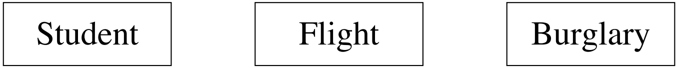
Entity: a *distinguishable* object

Entity set: set of entities of same type
Examples:

- students currently at University of Waterloo
- flights offered by Air Canada
- burglaries in Ontario during 1994

eg 2.
Student/2
Flight/2
Burglary/2
DB |= $\forall$e, n1, n2 (Student-student number(e, n1) $\wedge$ (Student-student number(e, n2) -> n1 = n2)
DB |= $\forall$e (Student(e)-> $\exists$n Student-studentnumber(e, n))

Graphical representation of entity sets:

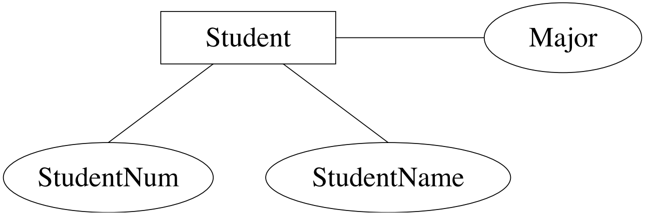


Basic E-R Modeling (cont'd)

Attributes: describe properties of entities
Examples (for Student-entities): StudentNum, StudentName, Major, . . .

Domain: set of permitted values for an attribute

Graphical representation of attributes:



Basic E-R Modeling (cont'd)

Relationship: representation of the fact that certain entities are related to each other

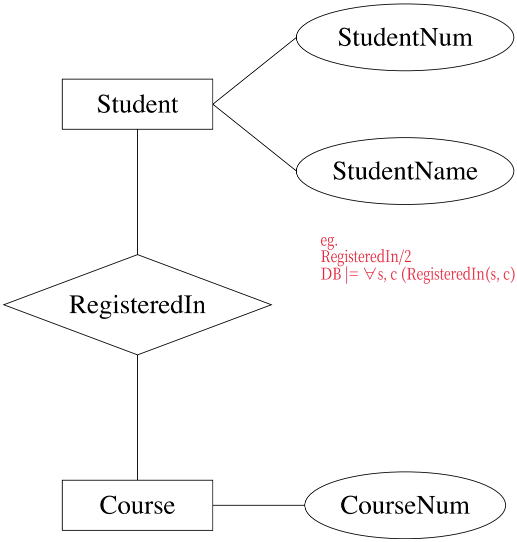
Relationship set: set of relationships of a given type

Examples:

- students registered in courses
- passengers booked on flights
- parents and their children
- bank branches, customers and their accounts

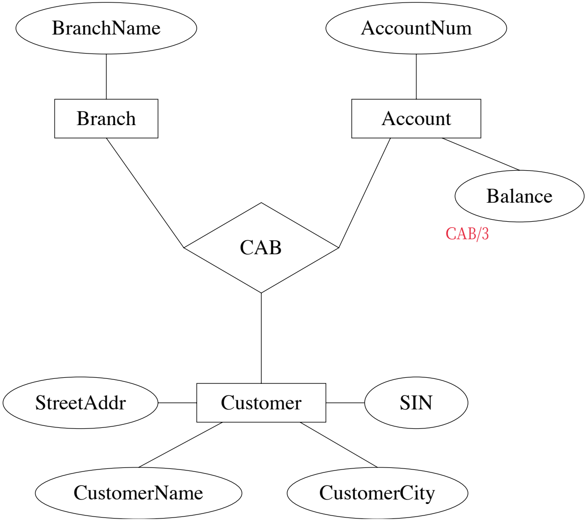
In order for a relationship to exist, the participating entities must exist.

Graphical Representation



eg.
RegisteredIn/2
DB $\models \forall s, c\ (\text{RegisteredIn}(s, c) \rightarrow (\text{Student}(s) \wedge \text{Course}(c))$

Graphical Representation (cont'd)



Multiple Relationships and Role Names

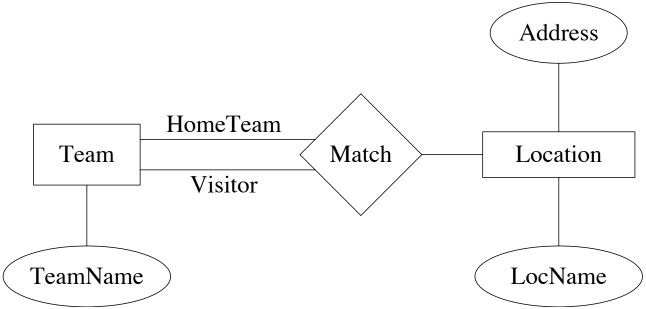
Role: the function of an entity set in a relationship set

Role name: an explicit indication of a role

Example:

Match/4

DB $\models \forall s, l, v, h$ Match(s, l, v, h) -> Location(l) $\wedge$ Team(v) $\wedge$ Team(h)



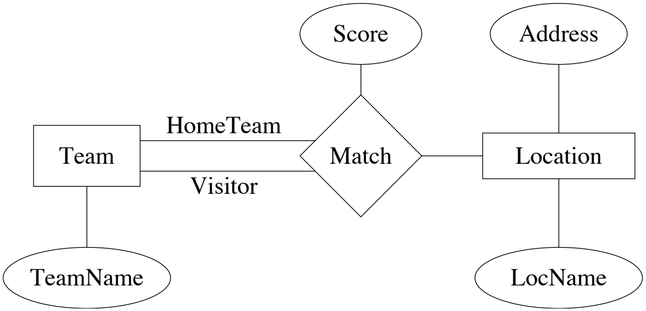
Role labels are needed whenever an entity set has multiple functions in a relationship set.

Relationships and Attributes

Relationships may also have attributes

Match/4
DB $\models \forall s1, s2, l, v, h\ (Match(s1, l, v, h) \wedge Match(s2, l, v, h)) \rightarrow s1 = s2$

Example:



Constraints in E-R Models

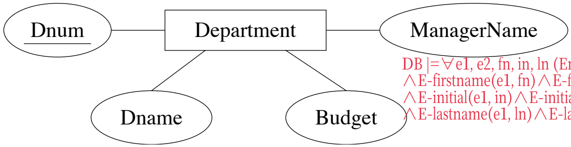
- Primary keys
- Relationship types
- Existence dependencies
- General cardinality constraints

Primary Keys

Each entity must be distinguishable from any other entity in an entity set by its attributes

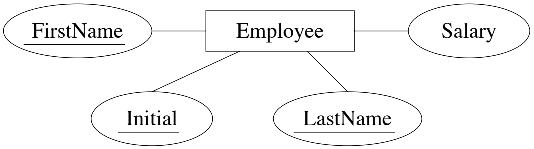
Primary key: selection of attributes chosen by designer values of which determines the particular entity.

Example 1:



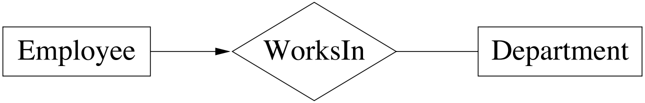
$$
\begin{aligned}
&DB \models \forall e1, e2, fn, in, ln\ (\text{Employee}(e1) \land \text{Employee}(e2)\\
&\land \text{E-firstname}(e1, fn) \land \text{E-firstname}(e2, fn)\\
&\land \text{E-initial}(e1, in) \land \text{E-initial}(e2, in)\\
&\land \text{E-lastname}(e1, ln) \land \text{E-lastname}(e2, ln)) \rightarrow e1=e2
\end{aligned}
$$

Example 2:

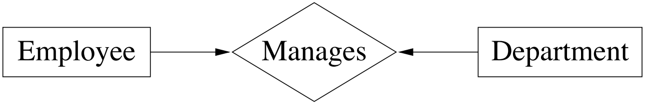


Relationship Types

- **many-to-many (N:N)**: an entity in one set can be related to many entities in the other set, and vice versa
  (This is the interpretation we have used so far.)
- **many-to-one (N:1)**: each entity in one set can be related to at most one entity in the other, but an entity in the second set may be related to many entities in the first



- **one-to-many (1:N)**: similar
- **one-to-one (1:1)**: each entity in one set can be related to at most one entity in the other, and vise versa



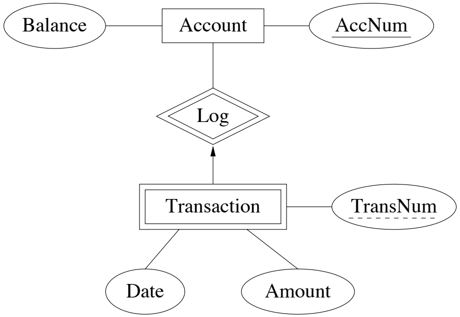
Existence Dependencies

Sometimes the existence of an entity depends on the existence of another entity

If *x* is **existence dependent** on *y*, then

- *y* is a **dominant entity**
- *x* is a **subordinate entity**

Example: “Transactions are existence dependent on accounts.”



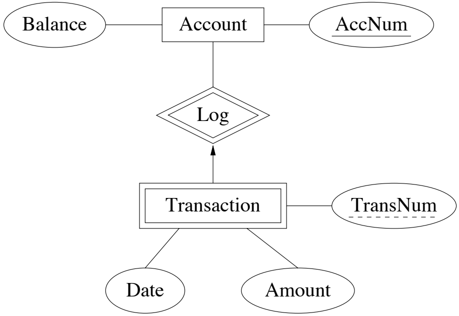
Identifying Subordinate Entities

Weak entity set: an entity set containing subordinate entities

Strong entity set: an entity set containing no subordinate entities

Attributes of weak entity sets only form key relative to a given dominant entity

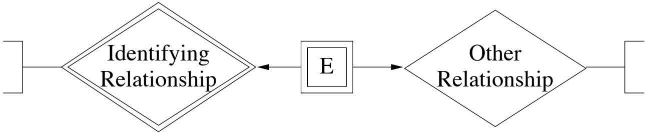
Example: "All transactions for a given account have a unique transaction number."



Identifying Subordinate Entities (cont'd)

A weak entity set must have a many-to-one relationship to a distinct entity set

Visualization: (distinguishing an identifying relationship)



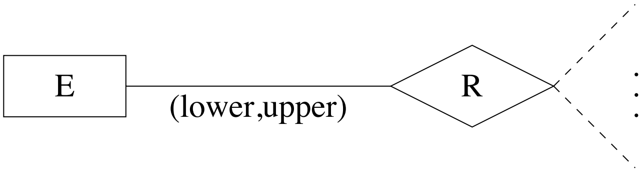
Discriminator of a weak entity set: set of attributes that distinguish subordinate entities of the set, for a particular dominant entity

Primary key for a weak entity set: discriminator + primary key of entity set for dominating entities

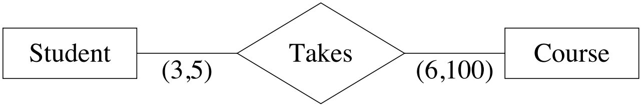
General Cardinality Constraints

General cardinality constraints determine lower and upper bounds on the number of relationships of a given relationship set in which a component entity may participate

Visualization:



Example:



Extensions to E-R Modeling

- Structured attributes
- Aggregation
- Specialization
- Generalization
- Disjointness

Structured Attributes

Composite attributes: composed of fixed number of other attributes

Multi-valued attributes: attributes that are set-valued

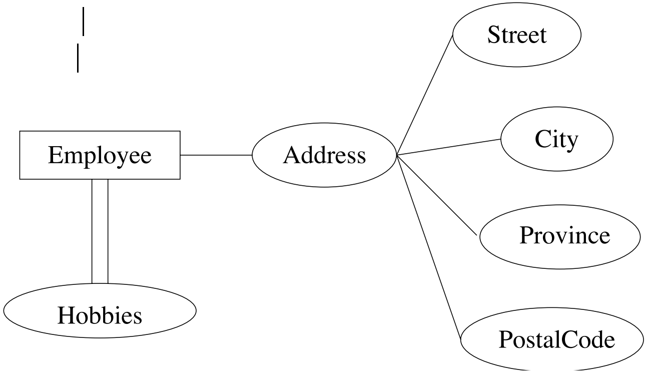
Example:

E-Hobbies

E	A

E-Address

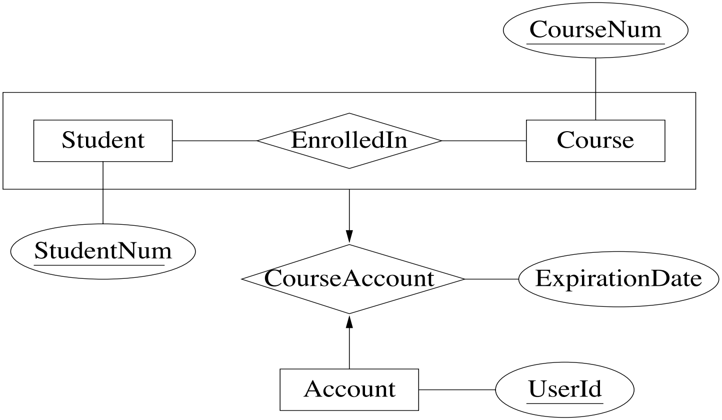
E	St	City	Prov	Post



Aggregation

Relationships can be viewed as higher-level entities

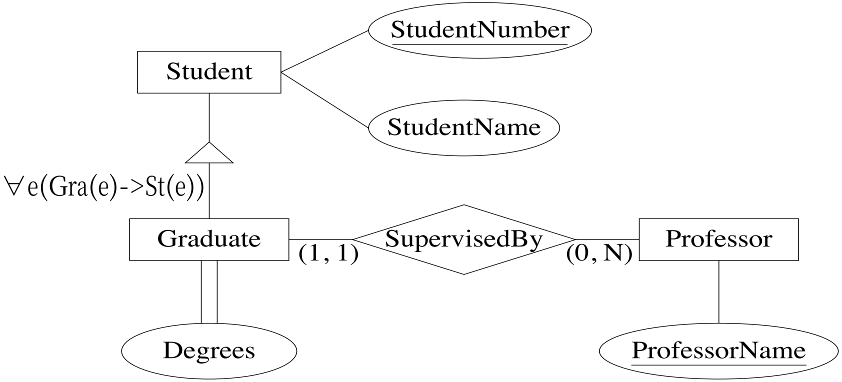
Example: “Accounts are assigned to a given student enrollment.”



Specialization

A specialized kind of entity set may be derived from a given entity set

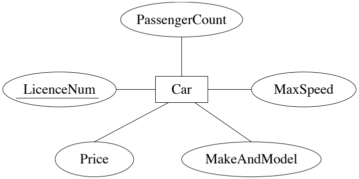
Example: “Graduate students are students who have a supervisor and a number of degrees.”

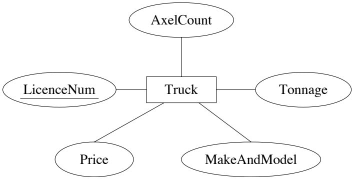


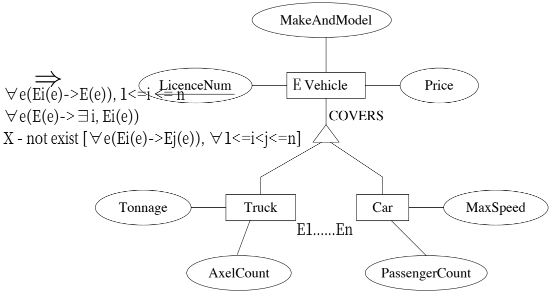
Generalization

Several entity sets can be abstracted by a more general entity set

Example: “A vehicle abstracts the notion of a car and a truck.”







$\Rightarrow$

$\forall e(Ei(e)\text{->}E(e)),\ 1\text{<=}i\text{<=}n$

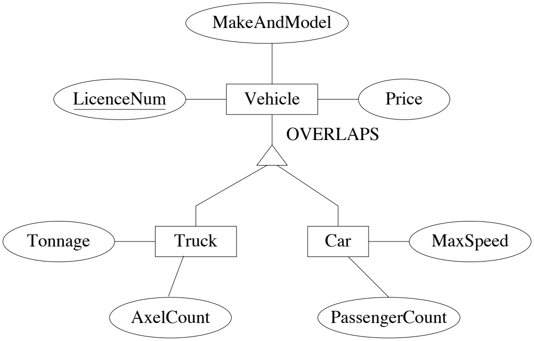
$\forall e(E(e)\text{->}\exists i,\ Ei(e))$

X - not exist $[\forall e(Ei(e)\text{->}Ej(e)),\ \forall 1\text{<=}i\text{<}j\text{<=}n]$

Disjointness

Specialized entity sets are usually disjoint but can be declared to have entities in common

- By default, specialized entity sets are disjoint.
  Example: We may decide that nothing is both a car and a truck.
- However, we can declare them to overlap (to accommodate utility vehicles, perhaps).



Designing An E-R Schema

Usually many ways to design an E-R schema
Points to consider

- use attribute or entity set?
- use entity set or relationship set?
- degrees of relationships?
- extended features?

Attributes or Entity Sets?

Example: Should one model employees' phones by a PhoneNumber attribute, or by a Phone entity set related to the Employee entity set?

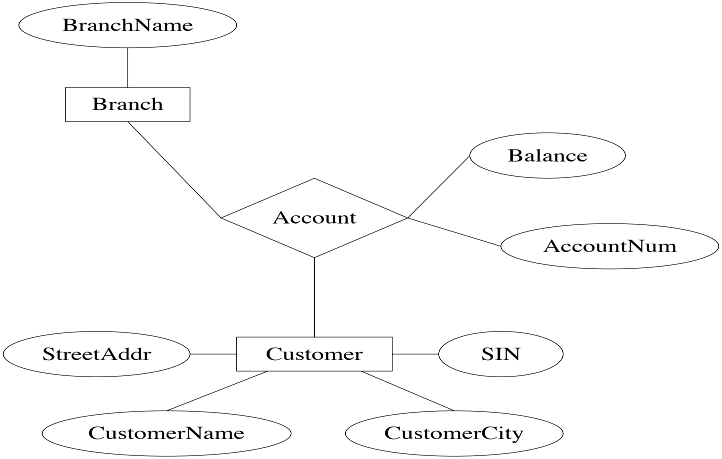
Rules of thumb:

- Is it a separate object?
- Do we maintain information about it?
- Can several of its kind belong to a single entity?
- Does it make sense to delete such an object?
- Can it be missing from some of the entity set's entities?
- Can it be shared by different entities?

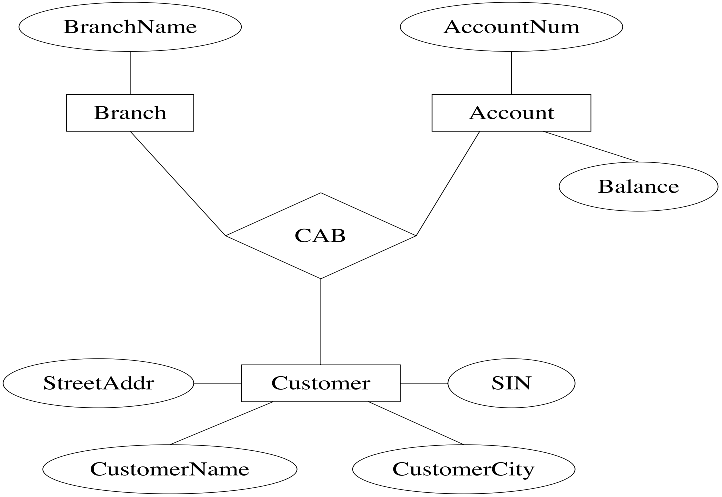
An affirmative answer to any of the above suggests a new entity set.

Entity Sets or Relationships?

Instead of representing accounts as entities, we could represent them as relationships

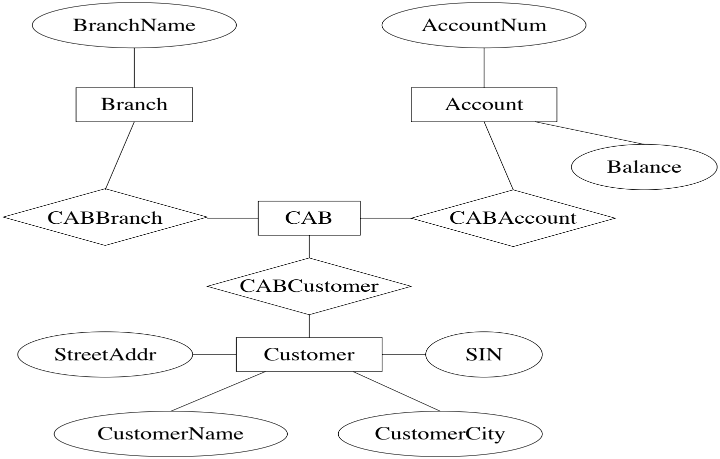


Binary vs. N-ary Relationships?



Binary vs. N-ary Relationships (cont'd)

We can always represent a relationship on *n* entity sets with *n* binary relationships



A Simple Methodology

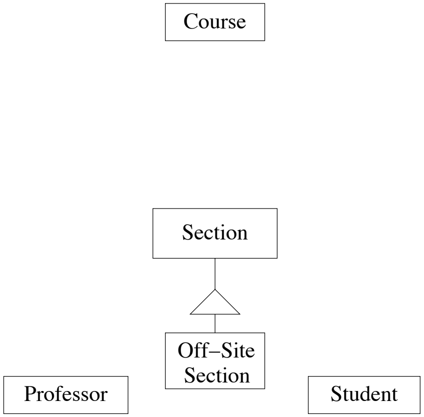
1. Recognize entity sets
2. Recognize relationship sets and participating entity sets
3. Recognize attributes of entity and relationship sets
4. Define relationship types and existence dependencies
5. Define general cardinality constraints, keys and discriminators
6. Draw diagram

For each step, maintain a log of assumptions motivating the choices, and of restrictions imposed by the choices

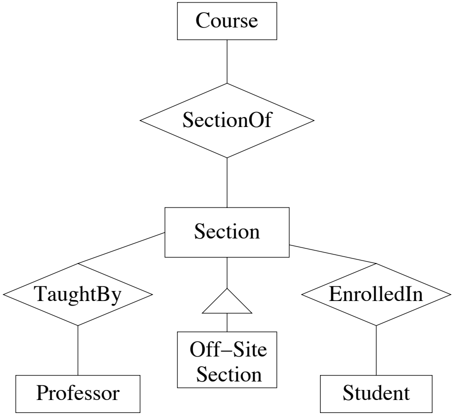
Example: A Registrar's Database

- Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.
- Most course sections are taught on-site, but a few are taught at off-site locations.
- Students have student numbers and names.
- Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.
- Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.
- A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

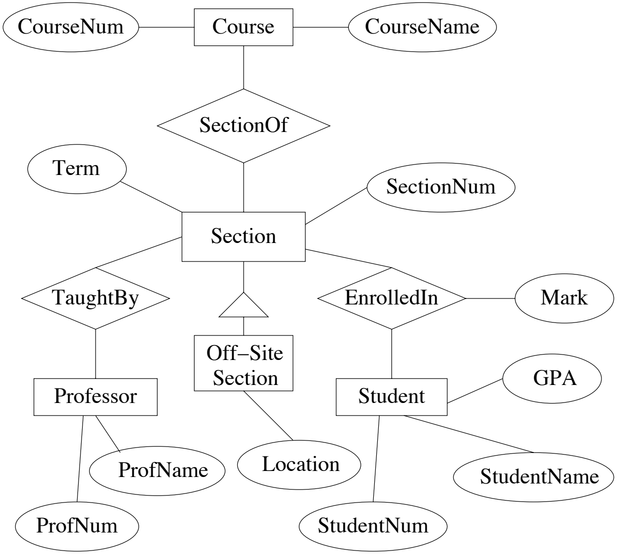
Example: A Registrar's Database (cont'd)



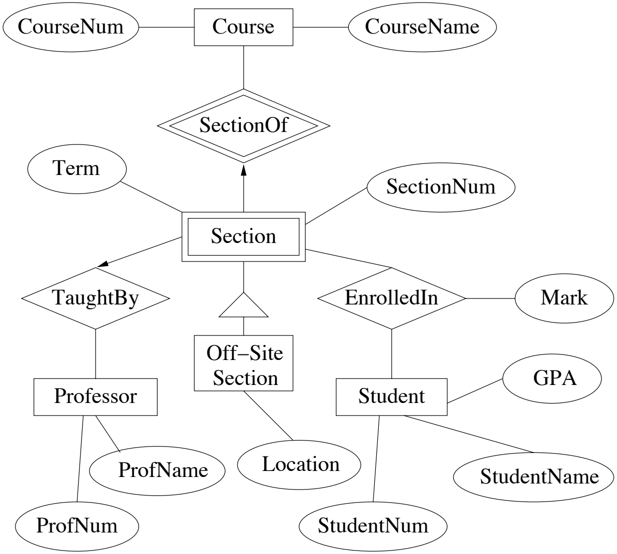
Example: A Registrar's Database (cont'd)



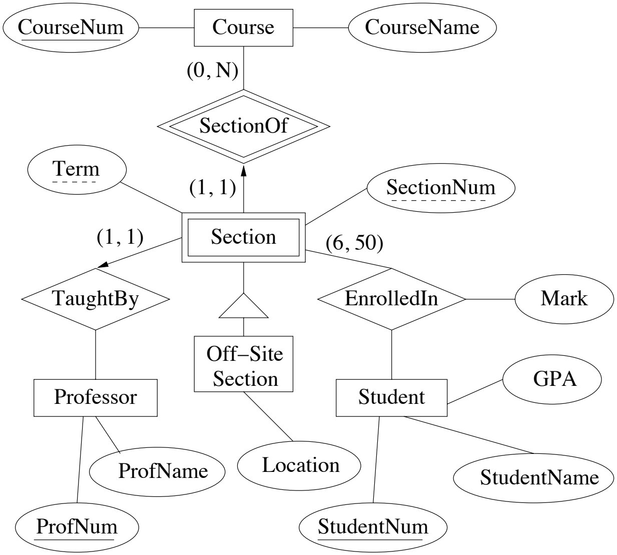
Example: A Registrar's Database (cont'd)



Example: A Registrar's Database (cont'd)



Example: A Registrar's Database (cont.'d)


CourseNum
Course
CourseName
(0, N)
SectionOf
Term
(1, 1)
SectionNum
(1, 1)
Section
(6, 50)
TaughtBy
EnrolledIn
Mark
Off–Site Section
GPA
Professor
Student
ProfName
Location
StudentName
ProfNum
StudentNum