CS 4604: Introduction to Database Management Systems

Virginia Tech CS 4604 Sprint 2021

Instructor: Yinlin Chen

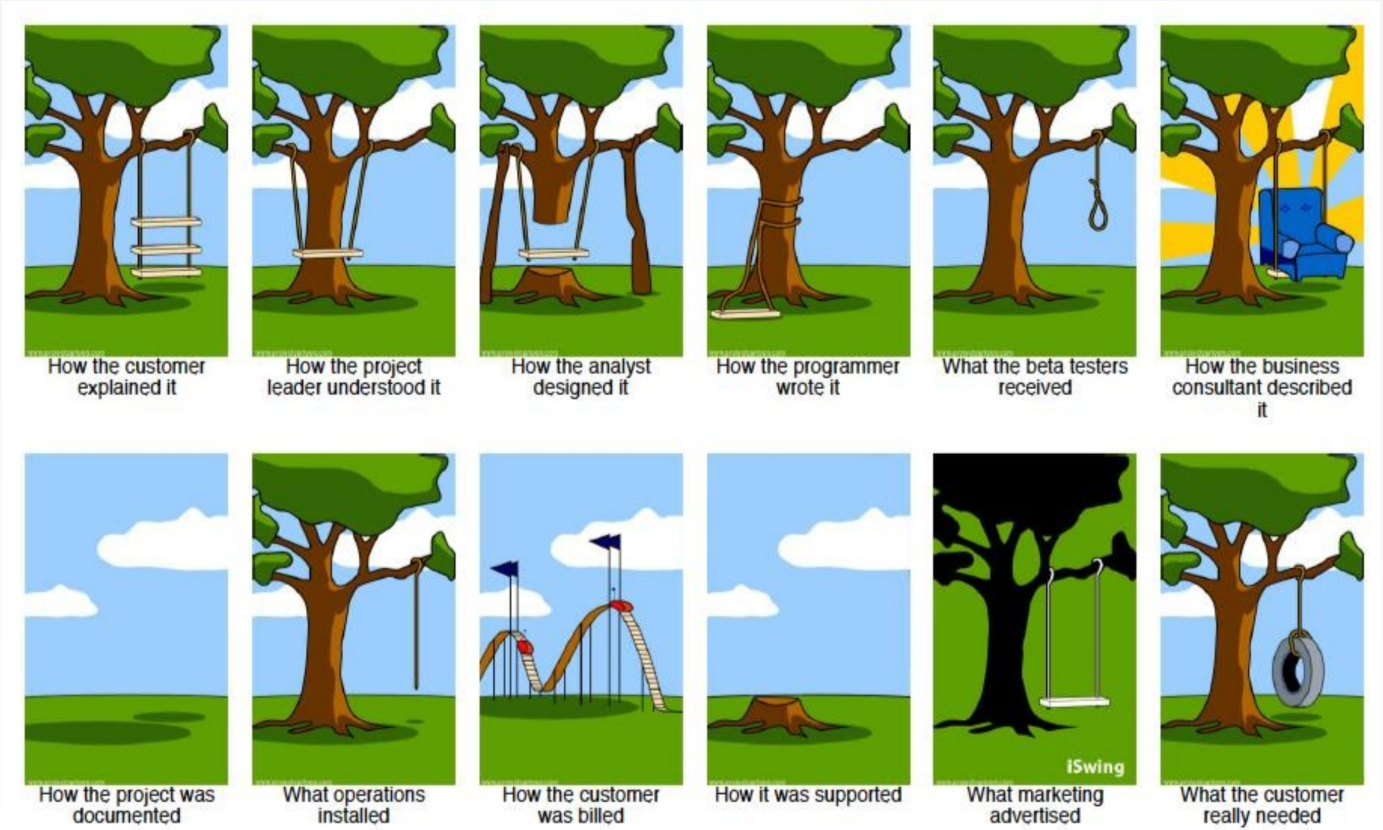
Today's Topics

- Database Design
- Entity-Relationship Models
- Enhanced Entity Relationship (EER) model

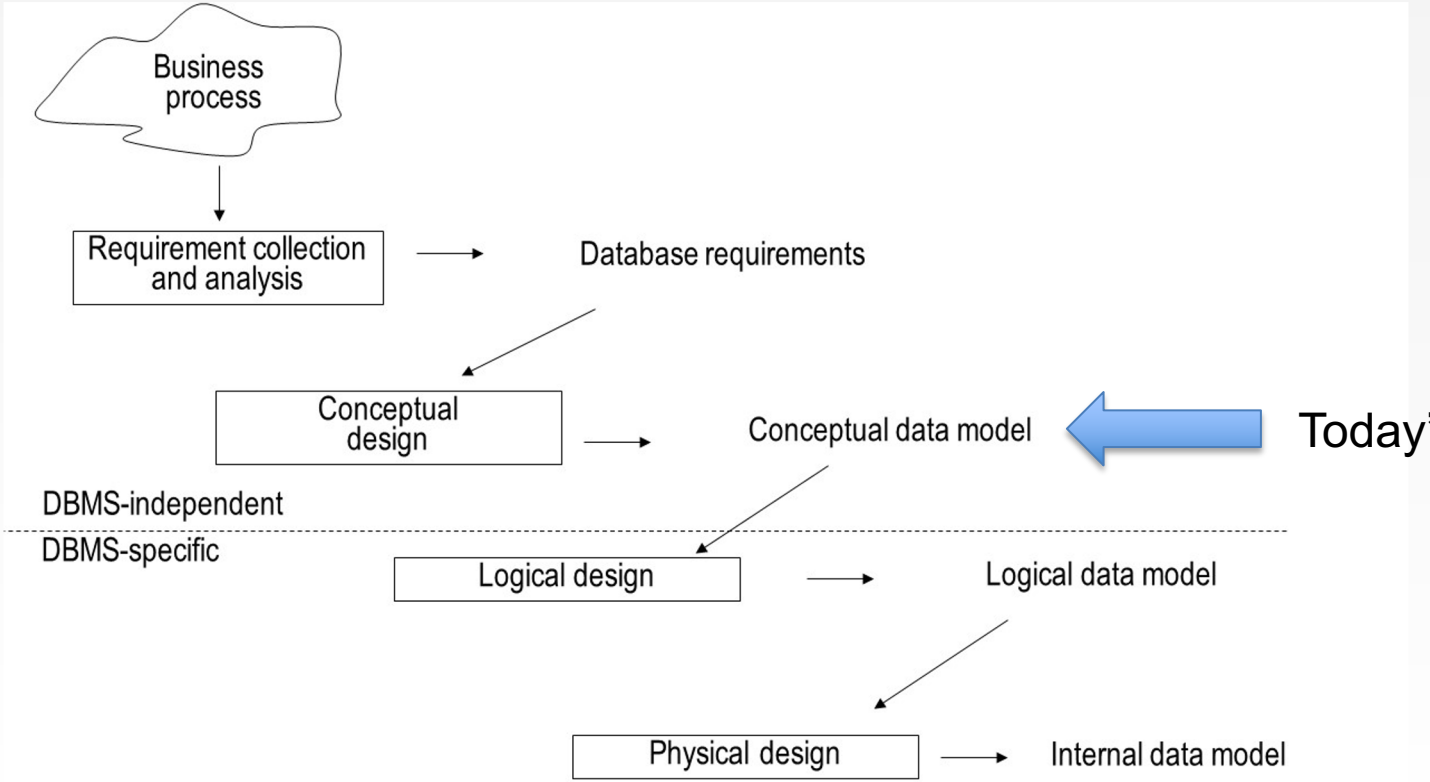
Important

- Follow only lecture slides for this topic!
- Slight differences from the textbook:
  - More details
  - Slightly different notation
  - Update/New content
  - Textbook published in 2002 ←

Software Project …



Phases of Database Design



Steps in Database Design

- **Requirements Analysis**
  - user needs; what must database do?
- **Conceptual Design**
  - *high level description (often done w/ER model)*
  - Object-Relational Mappings (ORMs: Hibernate, Rails, Django, etc) encourage you to program here
- **Logical Design**
  - translate ER into DBMS data model
  - ORMs often require you to help here too
- **Schema Refinement**
  - consistency, normalization
- **Physical Design** - indexes, disk layout
- **Security Design** - who accesses what, and how

Today's topics

Basic Database Terminology

- Data model : describes high-level conceptual structuring of data
  - Example: Data is set of student records, each with ID, name, address, and courses
  - Example: Data is a graph where nodes represent people and edges represent friendship relations
- Schema describes how data is to be structured and stored in a database
  - Schemas rarely change
- Data is actual “instance” of database
  - Updated continuously
  - Changes rapidly

Why Learn About Database Modeling?

- The way in which data is stored is very important for subsequent access and manipulation by SQL.
- Properties of a good data model:
  - It is easy to write correct and easy to understand queries.
  - Minor changes in the problem domain do not change the schema.
  - Major changes in the problem domain can be handled without too much difficulty.
  - Can support efficient database access.

Entity Relationship (ER) model

- An ER model is the result of **systematic analysis** to define and describe what is important to processes in an area of a business. It does not define the business processes; it only presents a **business data schema**
- A graph-based model
  - can be viewed as a graph, or a veneer over relations
  - “feels” more flexible, less structured

Purpose of ER Model

- A communication tool
  - The ER model allows us to sketch the design of a database informally
  - Represent different types of data and how they relate to each other
- Designs are drawings called *entity-relationship diagrams*.
- Fairly mechanical ways to convert ER diagrams to real implementations like relational databases exist.

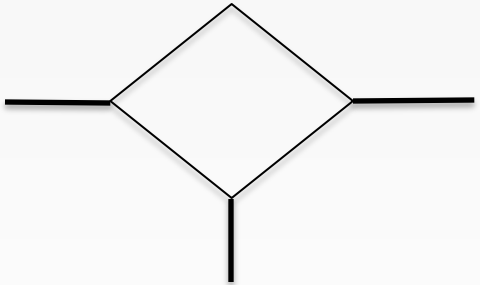
Purpose of ER Model

- When designing ER diagrams,
  - Forget about relations/tables
  - Only consider how to model the information you need to represent in your database
- Example
  - Professors advising students, Students taking courses, Students taught by professors

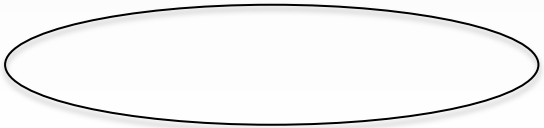
Tools

- Entities (‘entity sets’)

- Relationships (‘rel. sets’)
  and mapping constraints



- Attributes



Example

- Professors advising students, Students taking courses, Students taught by professors

Nouns → entity sets

Verbs → relationship sets

Entity and Entity Set

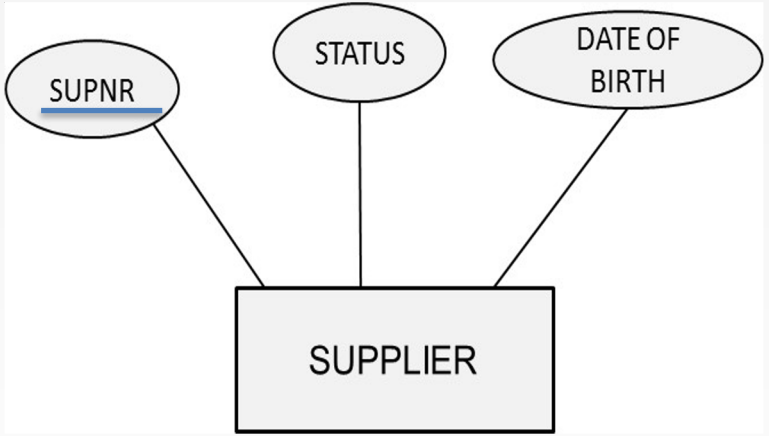
- Entity is one particular occurrence or instance of an entity set
  - A real-world object described by a set of attribute values
  - Examples: Sam Adams, Saranac and Guinness are entities from the entity set "beer”
- Entity set:
  - A collection of similar entities
  - All entities in an entity set have the same attributes
  - Each attribute has a domain
  - Examples: all students, all employees

Entity Sets

- *Entity* = “thing” or object
- *Entity set* = collection of similar entities.
  - Similar to a class in object-oriented languages.
- Attribute = property of an entity set.
  - Generally, all entities in a set have the same properties.
  - In most cases we use ‘atomic attributes’ e.g. integers, character strings etc.
  - There exist
    - **multivalued** or set-valued attributes (eg., ‘dependents’ for EMPLOYEE)
    - **derived** attributes (eg., 15% tip, age)

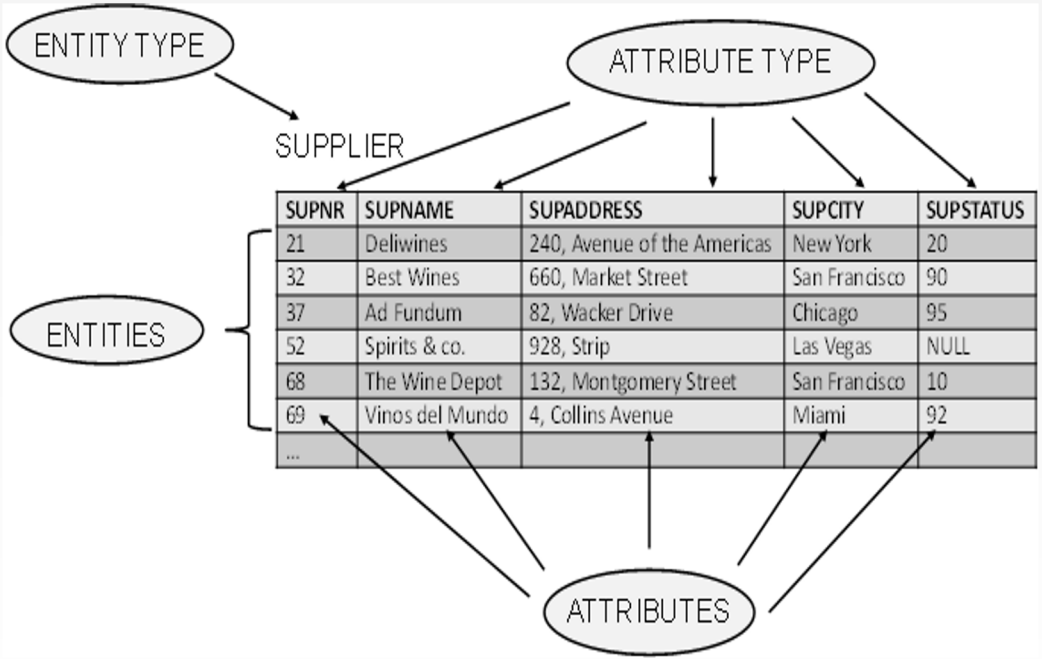
ER Diagrams

- In an ER diagram, each entity set is represented by a rectangle
- Each attribute of an entity set is represented by an oval, with a line to the rectangle representing its entity set



Attribute Types

- Attribute type represents a property of an entity set.
  - Example: supname and supaddress are attribute types of the entity set supplier
- Attribute is an instance of an attribute type



SUPNR	SUPNAME	SUPADDRESS	SUPCITY	SUPSTATUS
21	Deliwines	240, Avenue of the Americas	New York	20
32	Best Wines	660, Market Street	San Francisco	90
37	Ad Fundum	82, Wacker Drive	Chicago	95
52	Spirits & co.	928, Strip	Las Vegas	NULL
68	The Wine Depot	132, Montgomery Street	San Francisco	10
69	Vinos del Mundo	4, Collins Avenue	Miami	92
...				

Attribute Types

- Domains
- Key Attribute Types
- Simple versus Composite Attribute Types
- Single-Valued versus Multi-Valued Attribute Types
- Derived Attribute Type

Domains

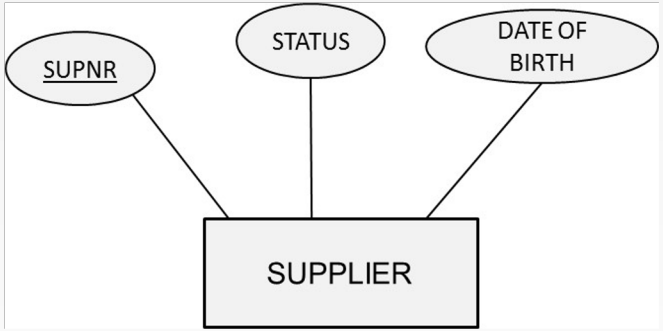
- A domain specifies the set of values that may be assigned to an attribute for each individual entity
  - Example: string, gender: {male and female}
- A domain can also contain null values
  - null value: value is not known, not applicable or not relevant
- Domains are not displayed in an ER model

(Annotations to your model are helpful, so does documentation)

Key Attribute Types

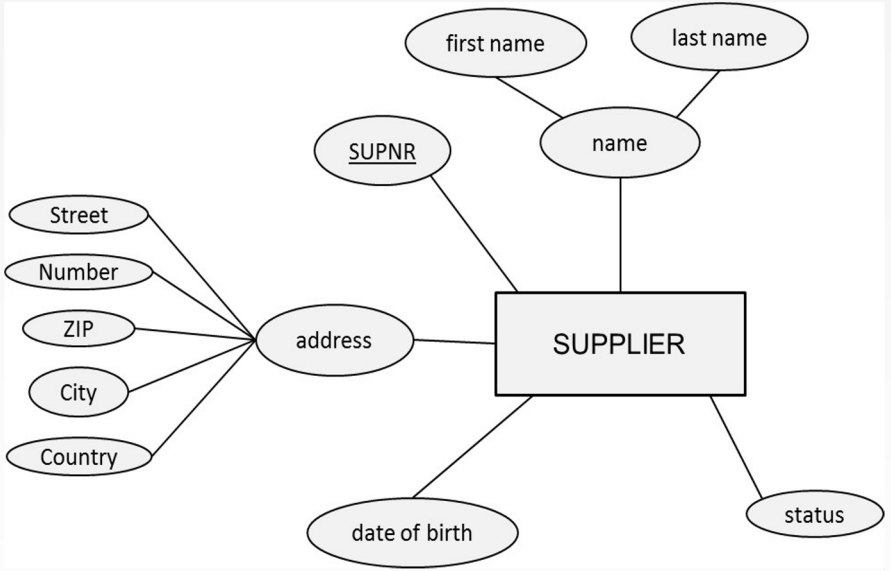
- A key attribute type is an attribute type whose values are distinct for each individual entity
  - Examples: supplier number, product number, social security number
- A key attribute type can also be a combination of attribute types
  - Example: combination of flight number and departure date

– Key attribute type is underlined



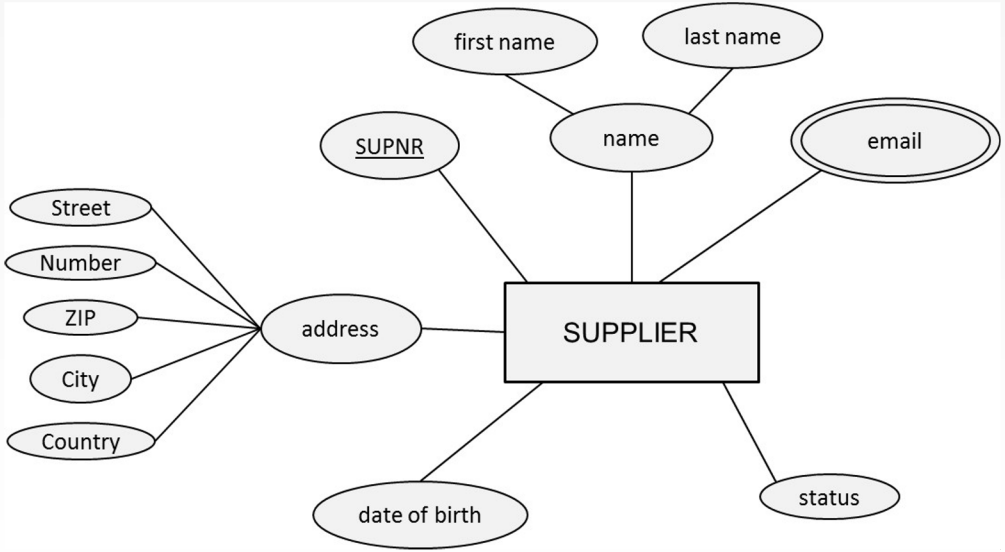
Simple versus Composite Attribute Types

- A simple or atomic attribute type cannot be further divided into parts
  - Examples: supplier number, supplier status
- A composite attribute type is an attribute type that can be decomposed into other meaningful attribute types
  - Examples: address, name



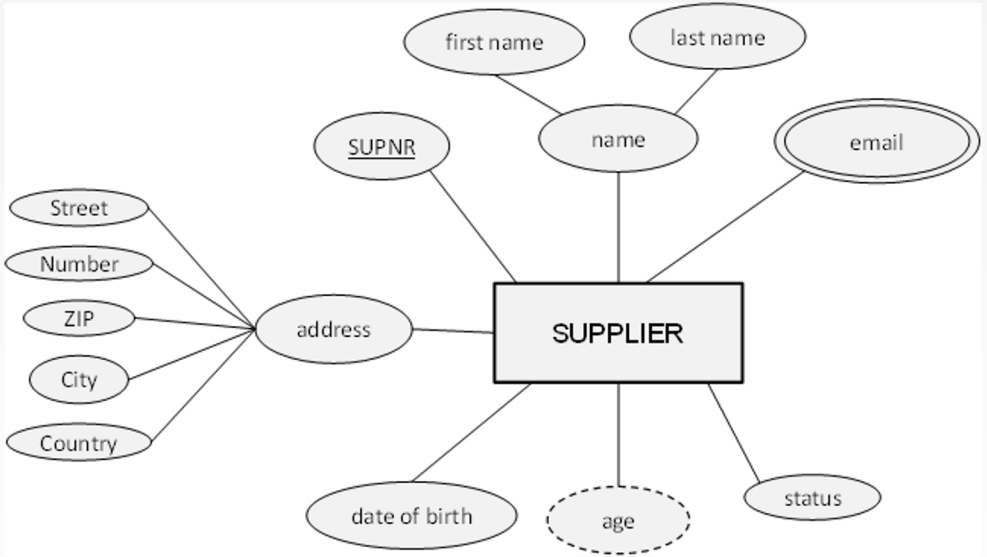
Single-Valued versus Multi-Valued Attribute Types

- A single-valued attribute type has only one value for a particular entity
  - Examples: date of birth, status
- A multi-valued attribute type is an attribute type that can have multiple values
  - Example: email



Derived Attribute Type

- A derived attribute type is an attribute type which can be derived from another attribute type
  - Example: age can be derived from date of birth



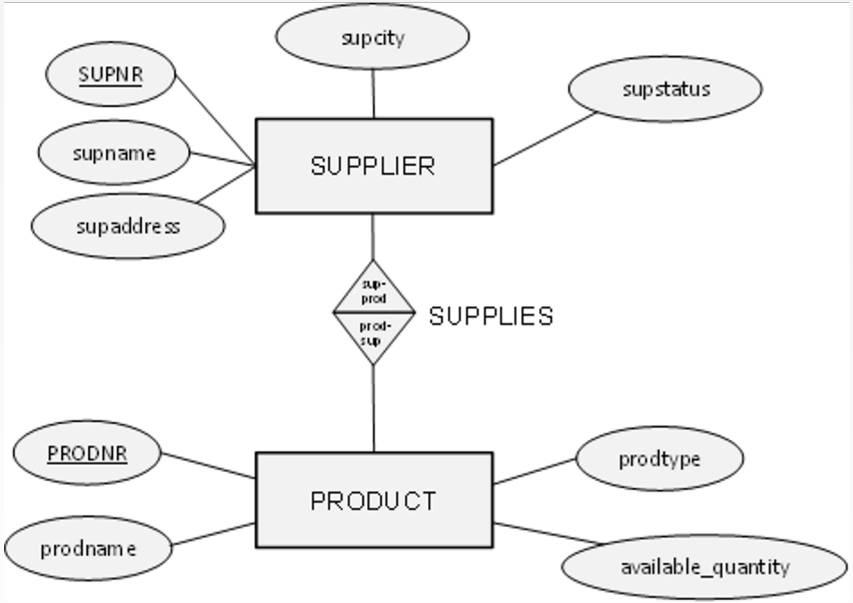
Instance of an Entity Set

- For each entity set, the instance stores a specific set of entities
- Each entity is a tuple containing specific values for each attribute
- Example: Instance of Entity set Students

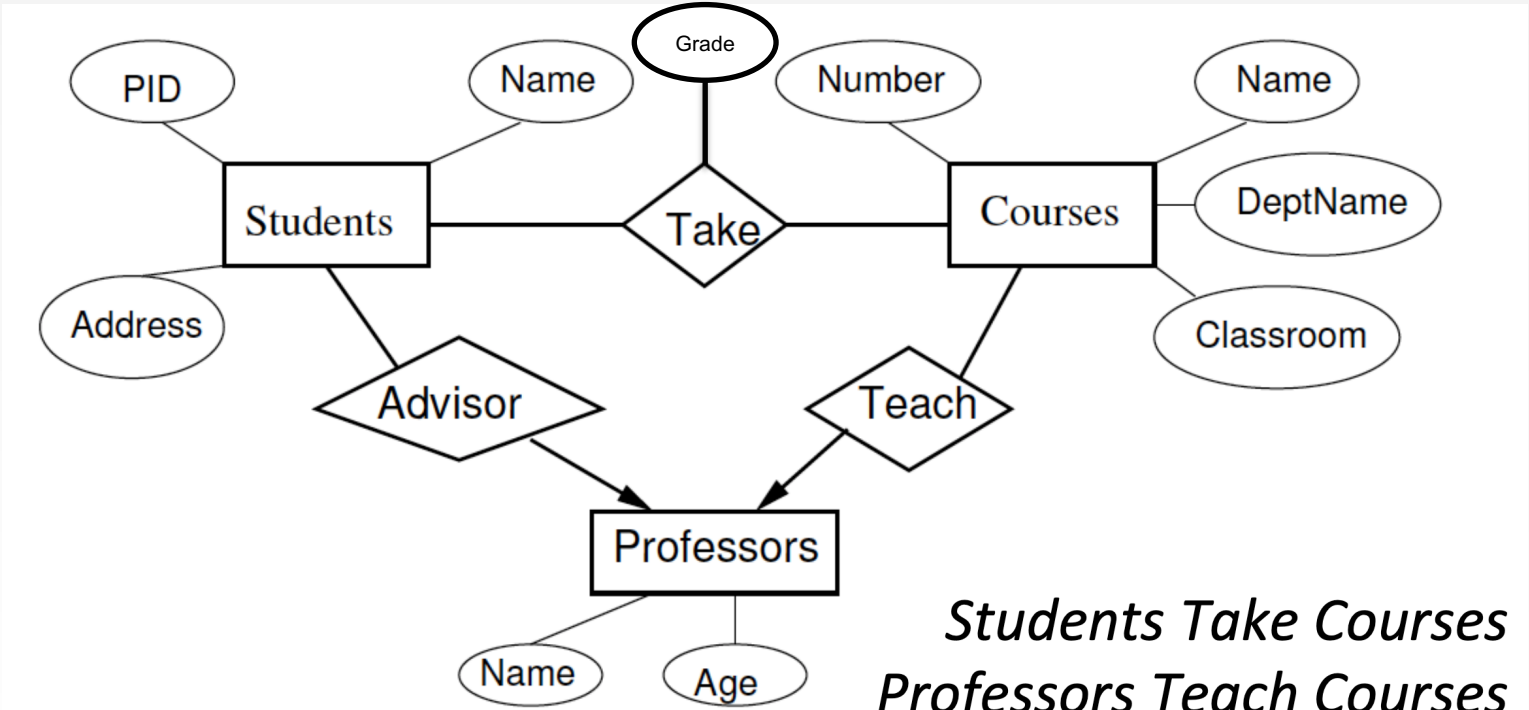
Name	*PID*	*Address*
Hermione Grainger	HG	Gryffindor Tower
Draco Malfoy	DM	Slytherin Tower
Harry Potter	HP	Gryffindor Tower
Ron Weasley	RW	Gryffindor Tower

Relationships

- A relationship connects two or more entity sets
- It is represented by a diamond, with lines to each of the entity sets involved
- A relationship set then defines a set of relationships among instances of one, two or more entity types
- Don't confuse 'Relationships' with 'Relations'!



Example: Relationships



Students Take Courses

Professors Teach Courses

Professors Advise Students

Instance of a Relationship

- Example: Instance of relationship Takes

StudentName	PID	Address	CourseName	Grade	DeptName
Hermione	HG	Gryffindor	Potions	A-	CS
Draco	DM	Slytherin	DefDrkArts	B	ECE
Harry	HP	Gryffindor	Potions	A	CS
Ron	RW	Gryffindor	Potions	C	CS

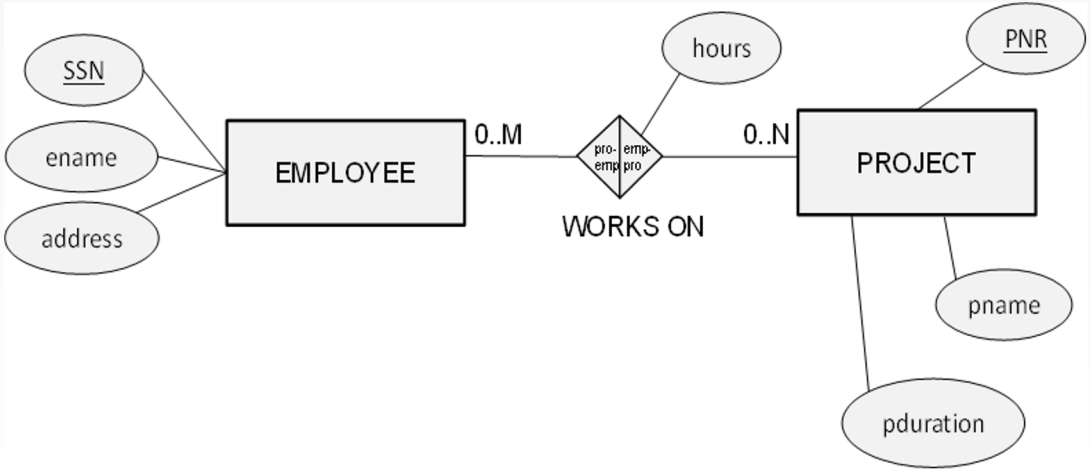
- Relationship R between (entity sets) E and F
  - Relates some *entities* in E to some *entities* in F

Instance of a Relationship

- Instance is a set of pairs of tuples (e; f) where e is in E and f is in F
  - Instance need not relate every tuple in E with every tuple in F
  - Relationship set R: the pairs of tuples (e; f) related by R
- (Conceptually) An instance of R is simply the ‘concatenation’ of the attribute lists for all pairs of tuples (e; f) in the relationship set for R
- ‘Tuples’ in R have two components, one from E and one from F

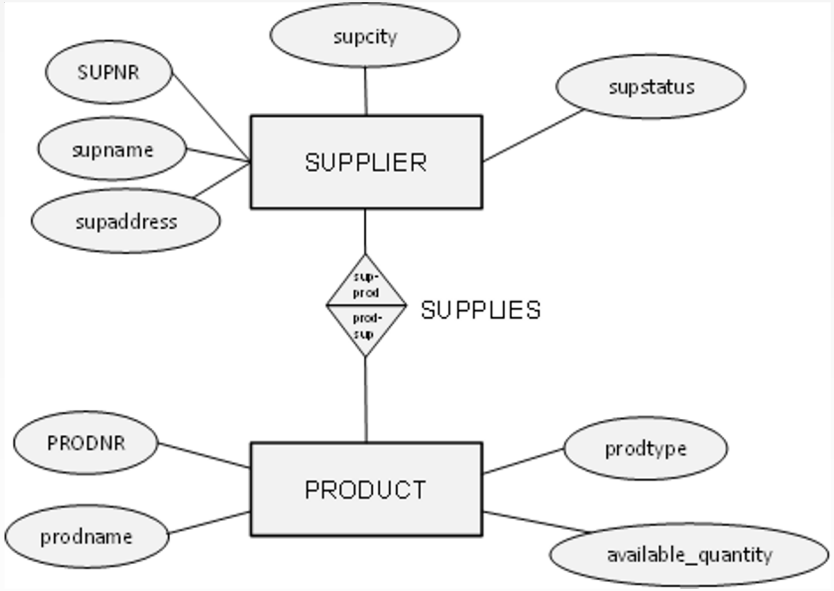
Relationship Attribute Types

- Relationship type can also have attribute types
- These attribute types can be migrated to one of the participating entity types in case of a 1:1 or 1:N relationship type (more on the next slides)

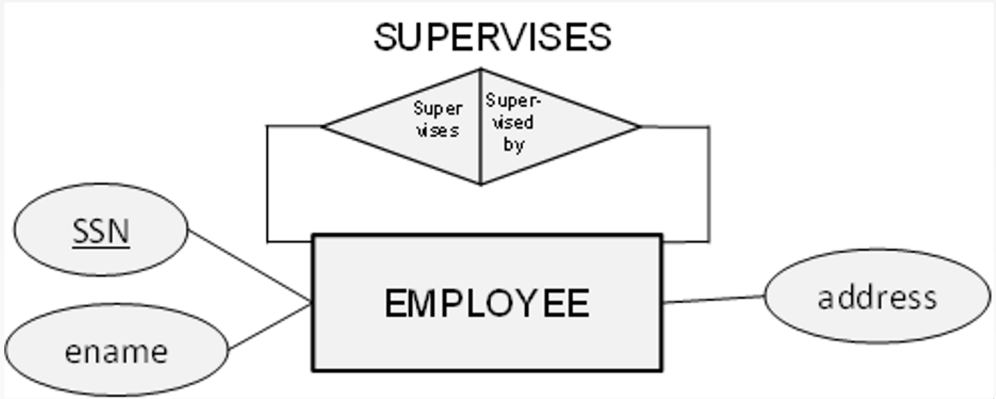


Degree and Roles

- The degree of a relationship type corresponds to the number of entity types participating in the relationship type
  - Unary: degree 1, binary: degree 2, ternary: degree 3
- The roles of a relationship type indicate the various directions that can be used to interpret it

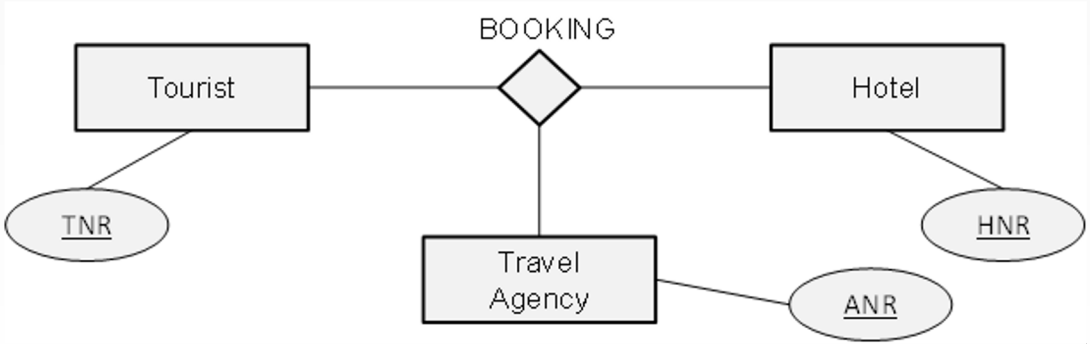


Degree and Roles



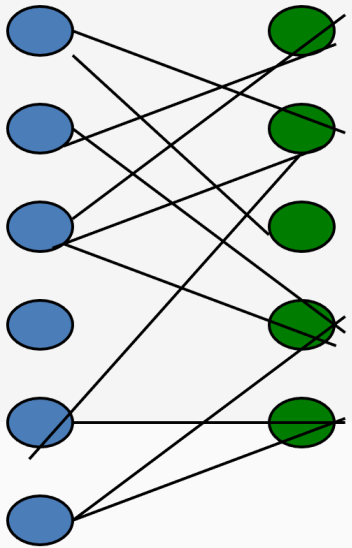
Unary

Ternary



Many-Many Relationships

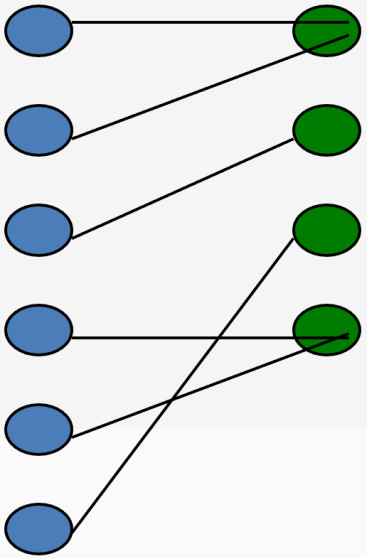
- In a *many-many* relationship, an entity of either set can be connected to many entities of the other set.



N:M

Many-One Relationships

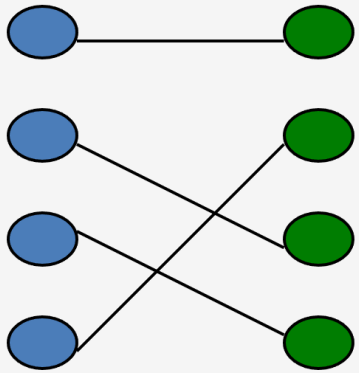
- Some binary relationships are *many-one* from one entity set to another
- Each entity of the first set is connected to at most one entity of the second set
- But an entity of the second set can be connected to zero, one, or many entities of the first set.



N:1

One-One Relationships

- In a one-one relationship, each entity of either entity set is related to at most one entity of the other set
- The schema defines the multiplicity of relationships. Don't use the instances of the schema to determine multiplicity.

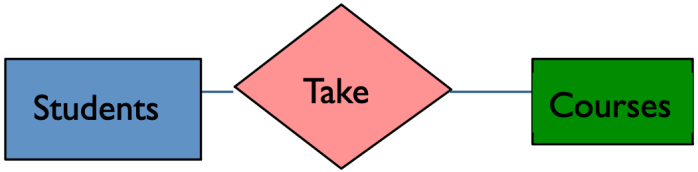


1:1

Representing “Multiplicity”

- Show a many-one relationship by an arrow entering the “one” side.
- Show a one-one relationship by arrows entering both entity sets.

many-one



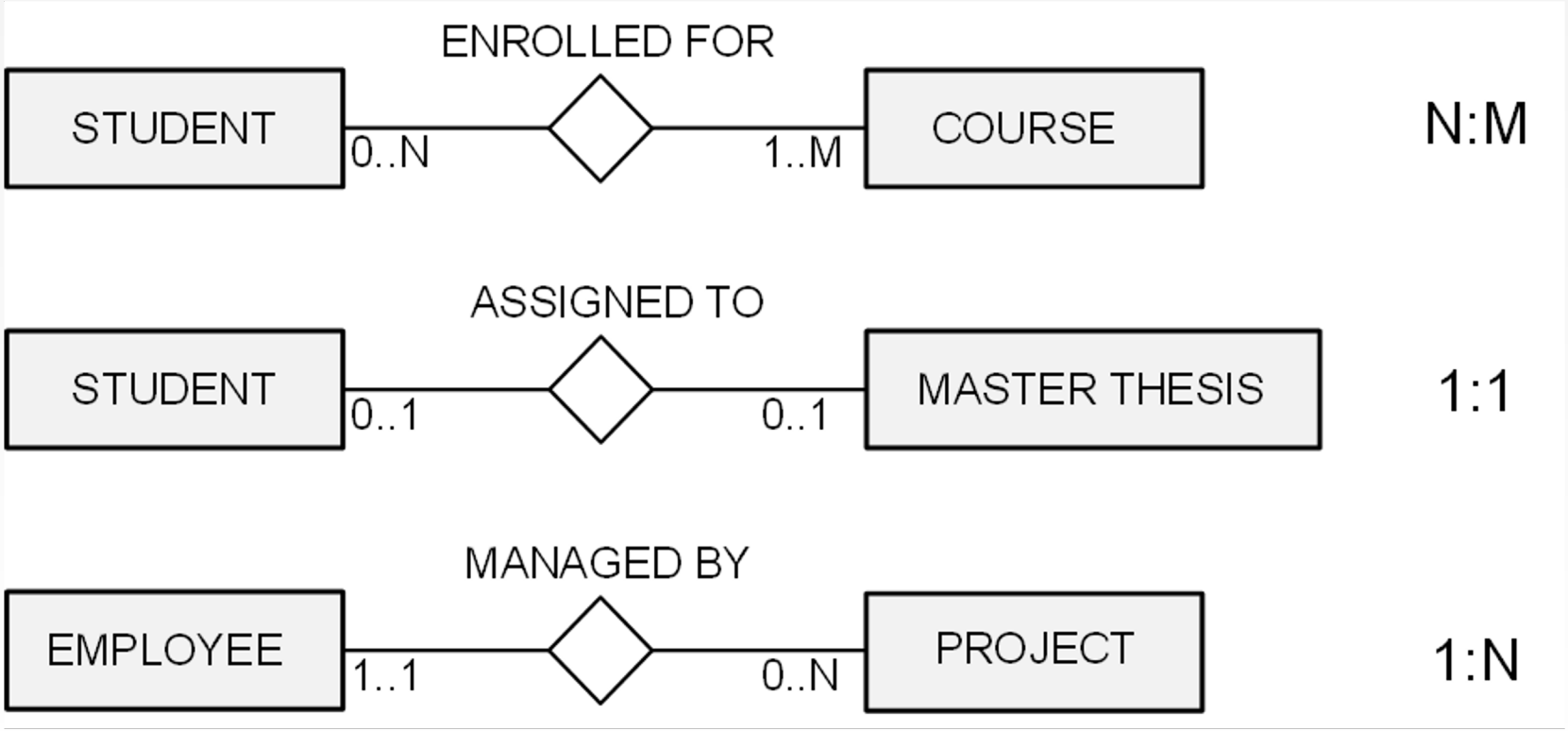
many-many

one-one

Cardinalities

- Every relationship type can be characterized in terms of its cardinalities, which specify the minimum or maximum number of relationship instances that an individual entity can participate in
- Minimum cardinality (participation) can be 0 or 1
  - If 0: partial participation
  - If 1: total participation or existence dependency
- Maximum cardinality can be 1 or N
- Relationship types are often characterized by their maximum cardinalities
  - 4 options for binary relationship types: **1:1, 1:N, N:1 and M:N.**

Cardinalities



Reading and Next Class

- Entity/Relationship Models I
  - Ch 2
- Next: Entity/Relationship Models II
  - ER to relational: Ch 3.5