CS 455
Database Management Systems

Est. 1888
UNIVERSITY of
PUGET
SOUND

Department of Mathematics
and Computer Science

Lecture 5
Entity Relationship Model

Disclaimer: Materials adapted for use from various sources

Topics

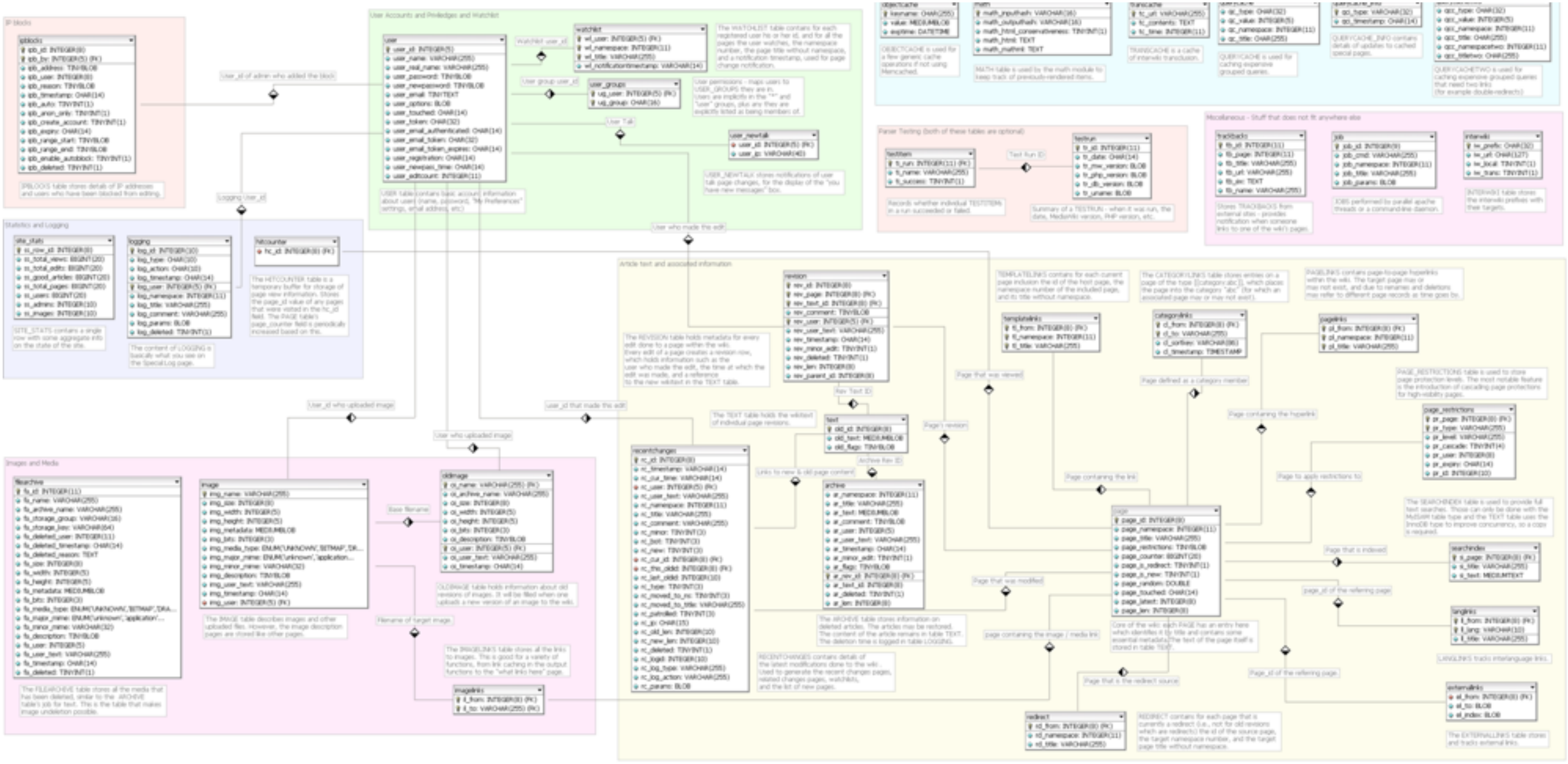
- Motivation
- The ER Building Blocks
  - Entity
  - Attributes
  - Relationships
  - Weak and Strong Entities
- Examples
- ER Reduction to Relations
- Conclusion

Motivation

- In the real world, DBs can have tons of relations!
- Here’s a snippet of Wikipedia’s DB schema
  - Problem: How did they arrive at this complicated schema?

```
page(page_id, page_namespace, page_title, page_restriction, ...)
redirect(rd_from, rd_namespace, rd_title)
externallinks(el_from, el_to, el_index)
searchindex(si_page, si_title, si_text)
page_links(pl_from, pl_namespace, pl_title)
category_links(cl_from, cl_to, cl_sortkey, cl_timestamp)
templatelinks(tl_from, tl_namespace, tl_title)
revision(rev_id, rev_page, rev_text_id, rev_comment, ...)
.
.
.
-- 36 total relations
```

Wikipedia DB Schema



Topics

- Motivation
- The ER Building Blocks
  - Entity and Entity Sets
  - Relationship and Relationship Sets
  - Attributes
  - Weak and Strong Entities
- Examples
- ER Reduction to Relations
- Conclusion

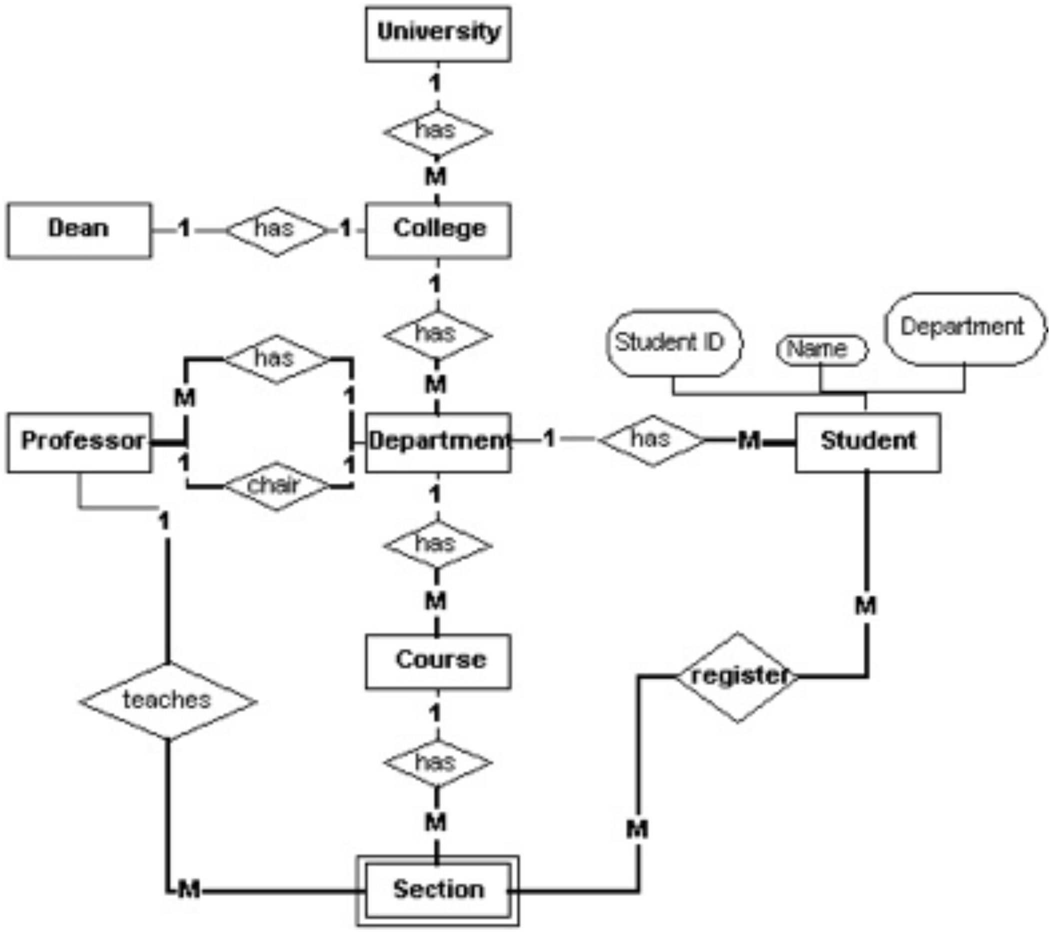
What Is the ER Model?

- The Entity-Relationship (ER) Model is a high-level way to *describe* an organization's database

- Goal
  - To describe data requirements and their relationships
  - *Like how UML is for modeling software before you start implementing your classes!*

What Is the ER Model? (Cont.)

- Create an accurate reflection of the real-world in a database!
- Basic assumption is database can be described as:
  - A collection of entities, and
  - The relationships among entities



Entity and Entity Sets

- An *entity* is a distinguishable object, thing, person for which the organization needs data
  - Concrete (*e.g.*, a person, a book)
  - Abstract (*e.g.*, a day, a sales transaction)
  - In general, entities are *nouns*

- An entity is an instance of a relation
  - Like classes vs. objects
  - For a pet store, some entities might be?
    - Bob, the customer
    - Fluffy, the snake

Entity and Entity Sets (Cont.)

- An *entity set* is a collection of entities that share a common definition
  - No longer ‘a’ person, or ‘a’ book, but instead:
    - The *set* of all customers
    - The *set* of pets for sale

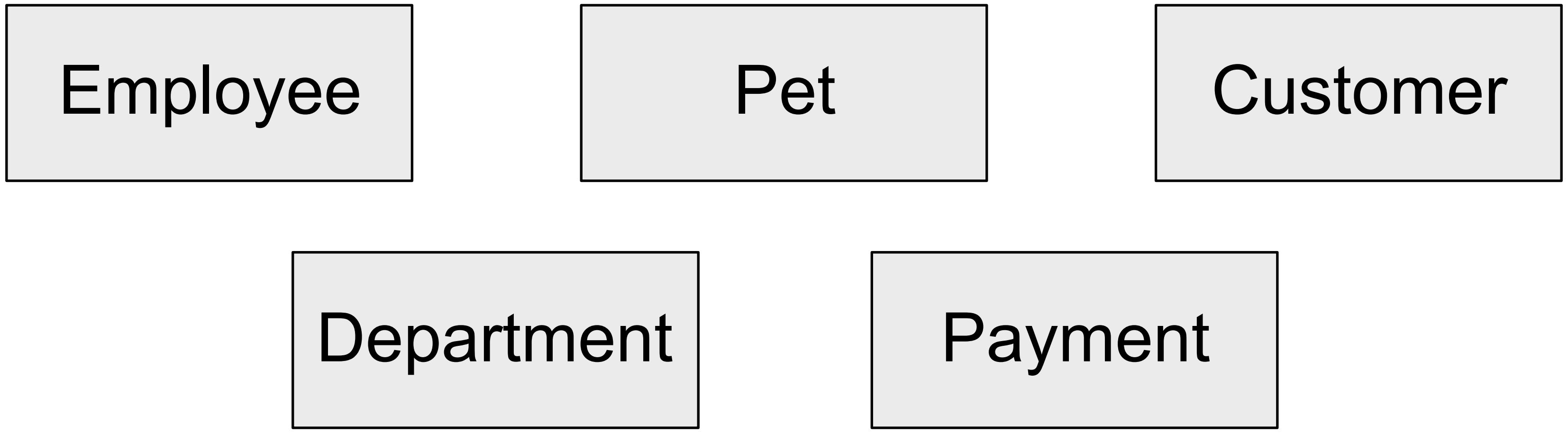
- Formally,
  - An entity set $E = \{e_1, \ldots, e_n\}$ where $e_i \in E$ is an entity

Entity and Entity Sets (Cont.)

- In an ER diagram, an *entity set* is represented with a rectangle.
- Let's model the entity sets of a Pet Store
  - Ask yourselves: "What are all the *things* I want to store data about?"

- *ER Diagram Syntax:* Entity Set

- For a simple Pet Store, these entity sets might suffice:



Topics

- Motivation
- The ER Building Blocks
    - Entity and Entity Sets
    - Relationship and Relationship Sets
    - Attributes
    - Weak and Strong Entities
- Examples
- ER Reduction to Relations
- Conclusion

Relationships and Relationship Sets

- A *relationship* is an association between two or more entities

- Formally,
  - If $E_1, E_2, \ldots, E_m$ are entity sets, then a *relationship* is defined

    $(e_1, e_2, \ldots, e_m)$ such that $e_1 \in E_1, e_2 \in E_2, \ldots, e_m \in E_m$

- Think of it as the *verb* between two or more entities
  - Gaby *owns* Fido
  - Michael *makes* Payment
  - Wes *works-in* Fish Department
  - Anna *manages* Reptile Department

Relationships and Relationship Sets (Cont.)

- A *relationship set* is a collection of relationships of the same type
  - *Gaby owns Fido* is one relationship and *Kyle owns Fluffy* is another
  - A relationship set would be the set containing all such *customer-owns-pet* relationships

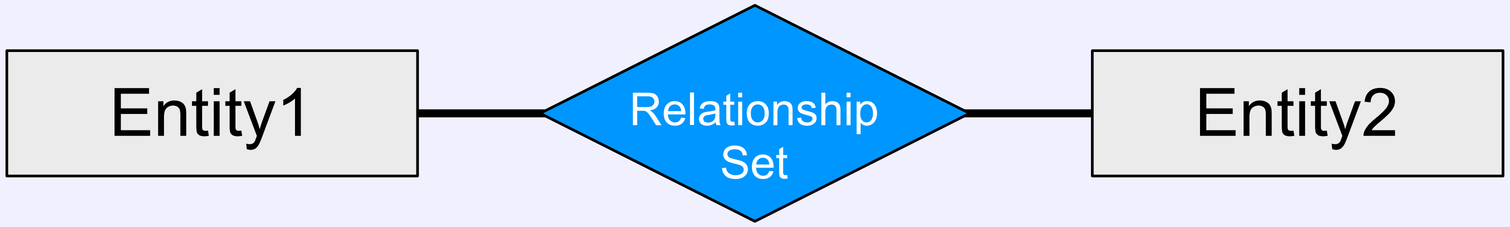
- Formally,
  - If $E_1, E_2, \ldots, E_m$ are entity sets, then a relationship set R is:

$$R \subseteq \{(e_1, e_2, \ldots, e_m) | e_1 \in E_1, e_2 \in E_2, \ldots, e_m \in E_m\}$$

where $(e_1, e_2, \ldots, e_m)$ is a relation.

Relationships and Relationship Sets (Cont.)

- ER Syntax:



- Let's assume we have two entity sets, Customer and Pet
  - With the following entities:

$$Customer = \{\text{Corey, Clarissa, Marea, Ben}\}$$

$$Pet = \{\text{Fluffy, Fido, Ferdi}\}$$

- Then the *relationship set, owns,* might look like this:

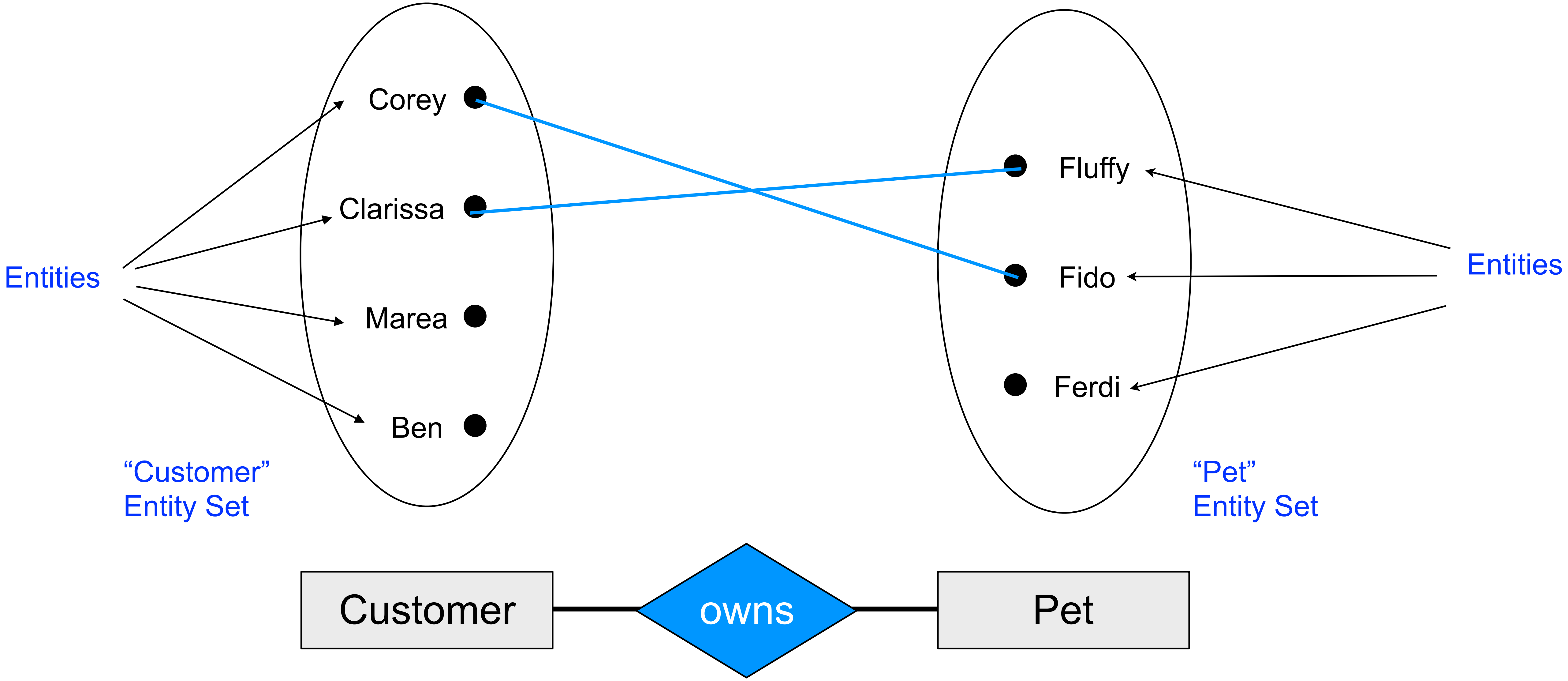
$$owns = \{(\text{Corey, Fido}), (\text{Clarissa, Fluffy})\}$$

Each tuple is a relationship between a *customer* and a *pet*

Visualizing Relationship Sets

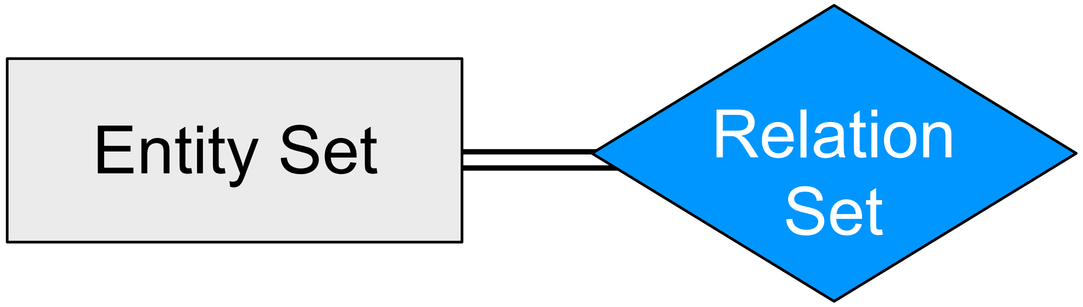
- Here's how to visualize what's going on:

$$owns = \{(\text{Corey, Fido}), (\text{Clarissa, Fluffy})\}$$

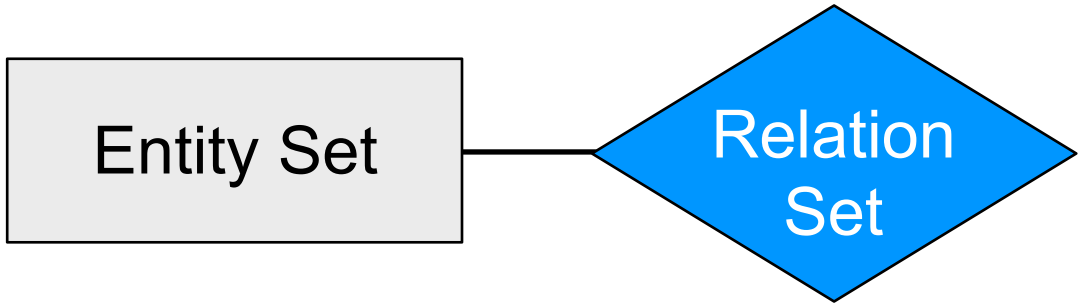


Entity-Relationship Participation Constraints

- *Participation Constraints*
  - Total Participation: Every entity in the entity set participates in at least one relationship in the relationship set.
    - Denoted by a double-line

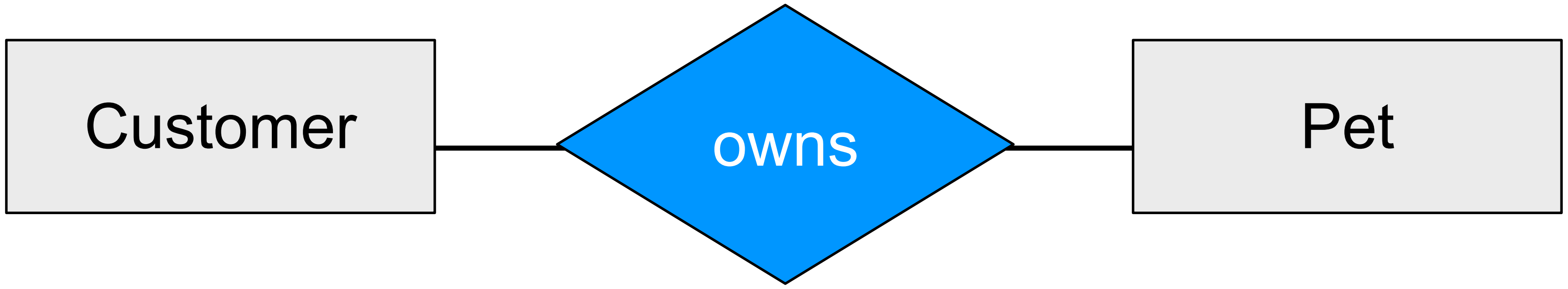


  - Partial Participation: Otherwise
    - Denoted by a single-line



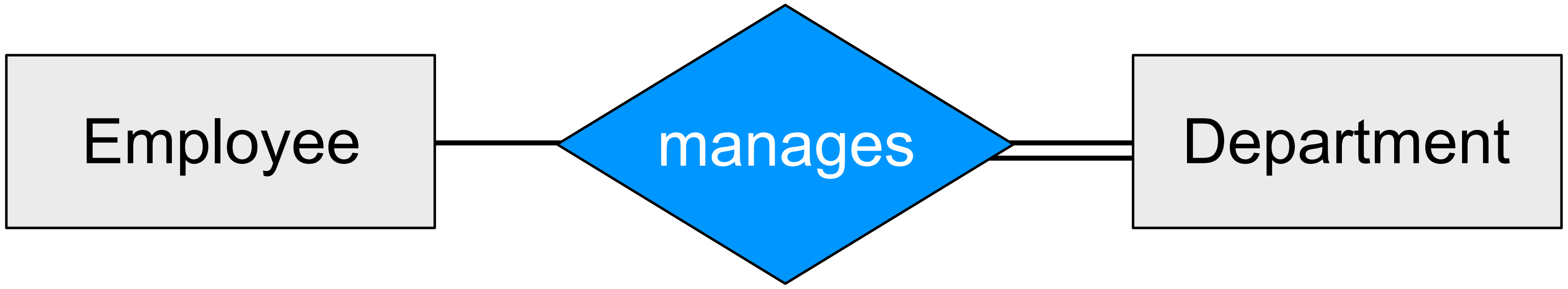
Participation Example

- Examples:



What it means: A customer does not necessarily have to own a pet! (Not all customers will participate in the “Owns” relation)

Similarly, not every pet is owned by a customer!



What it means: Not all employees manages a department.

However, every department is managed by at least one employee. (Total participation of department in the “Manages” relation.)

Entity-Relationship Mapping Cardinality

- *Mapping Cardinalities* express the number of entities to which another entity is linked by a relationship
  - 1-to-1: Read “one to one”
  - 1-to-N (or N-to-1): Read “one to many” or “many to one”)
  - M-to-N: Read “many to many”

1-to-1 Cardinality

- `1-to-1`: One entity in R is linked with at most one entity in S and vice versa.

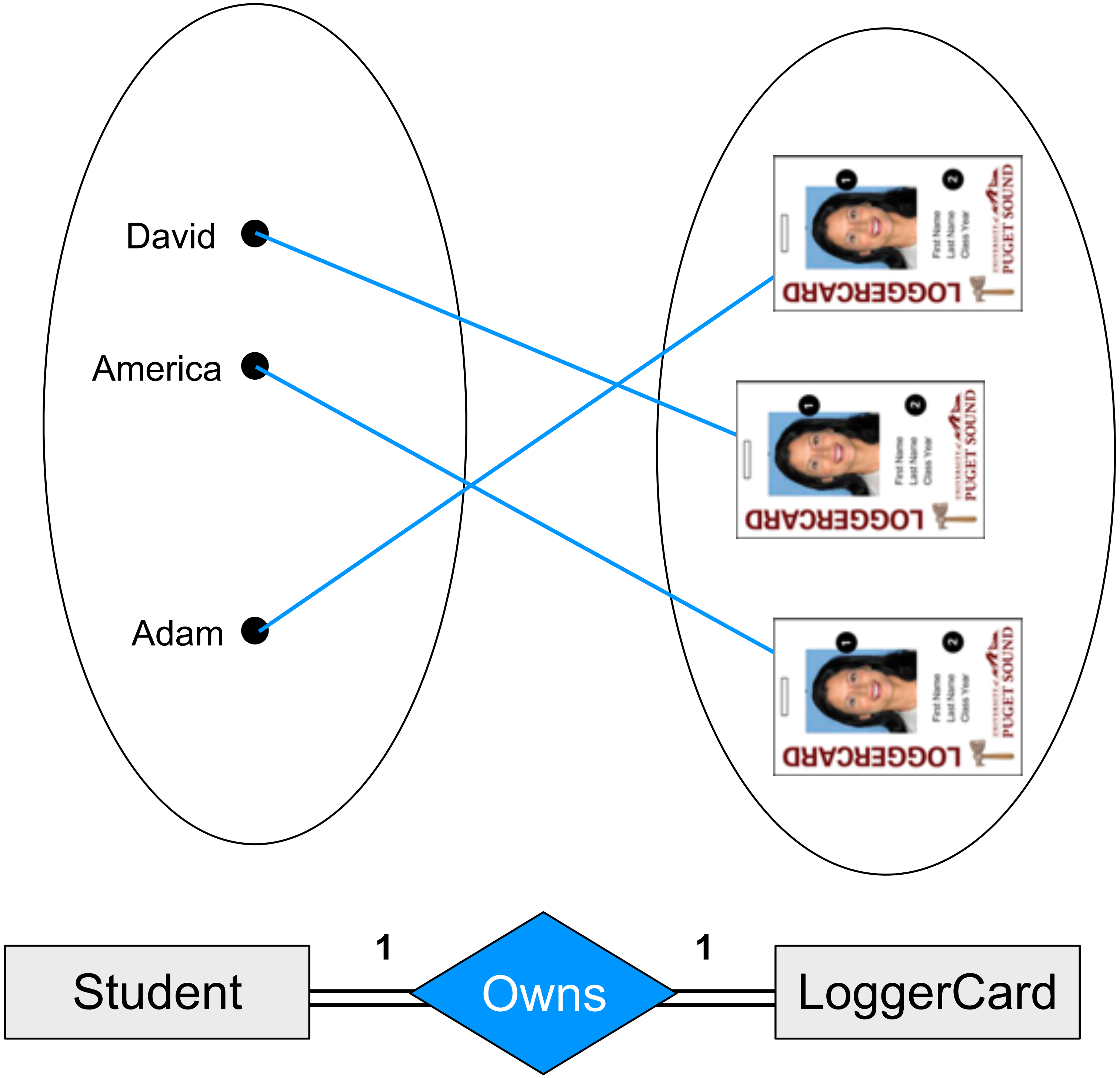
Real-World Assumptions:

All students must have a LoggerCard

Students are not allowed to own more than one LoggerCard

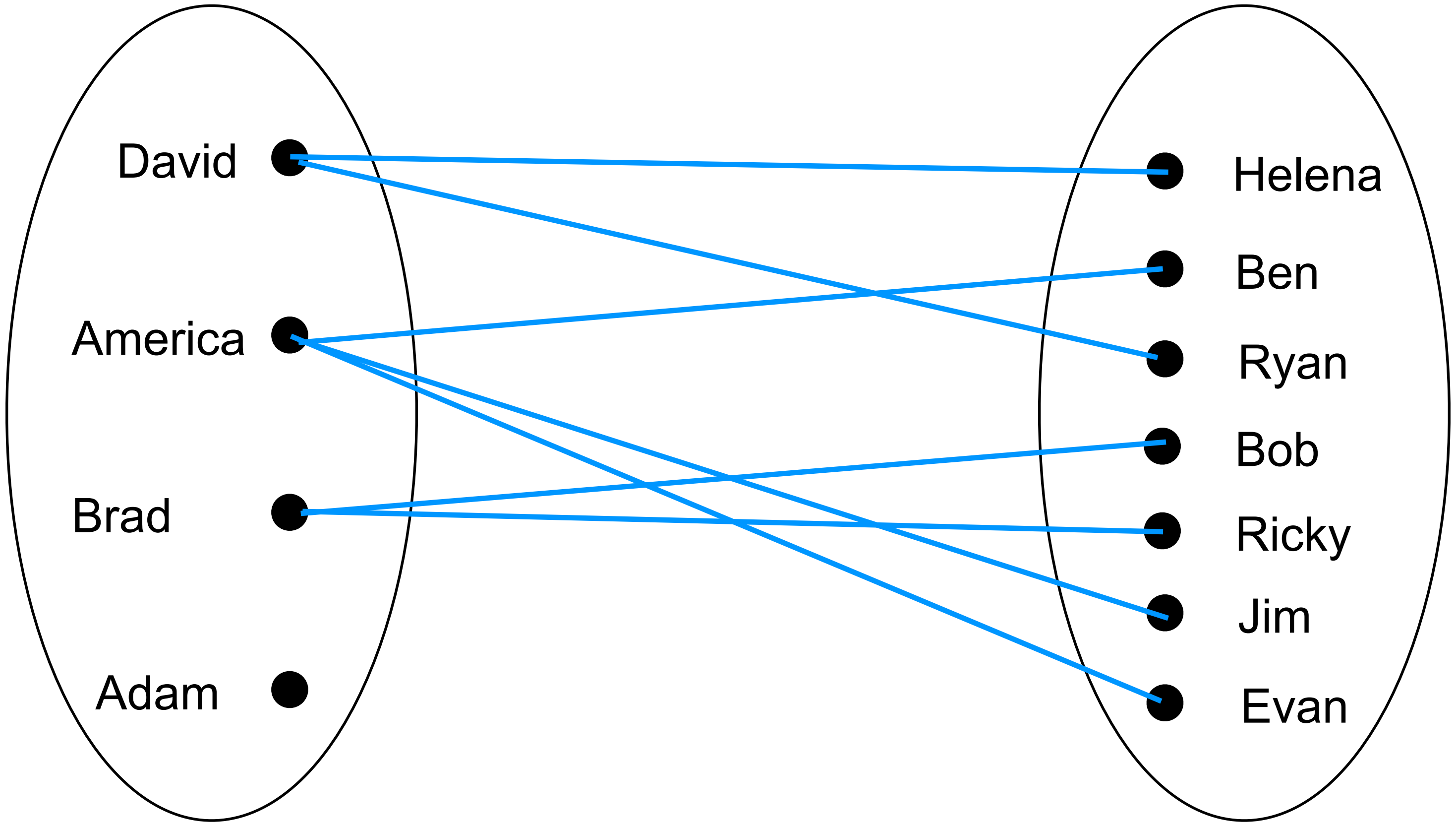
Students are not allowed to share a single LoggerCard

A LoggerCard cannot be in circulation if it's not owned by a student



1-to-N Cardinality

- `1-to-N` (or `N-to-1`): One entity in R is linked with N entities in S. Entities in S are linked with at most one entity in R.

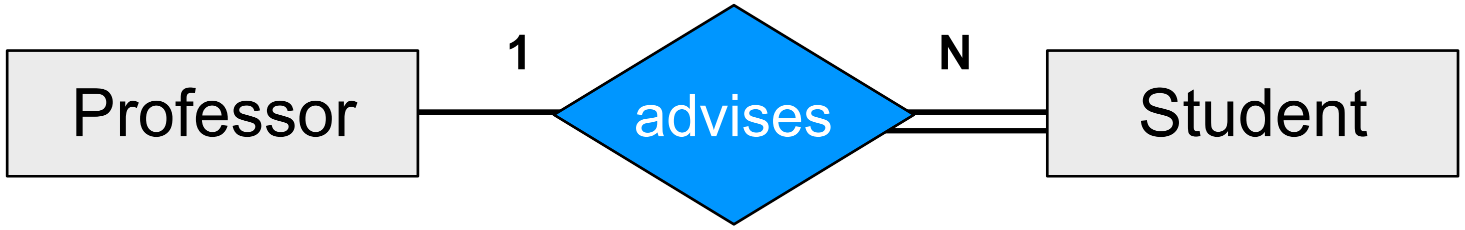


Real-World Assumptions:

Students must have at most one advisor

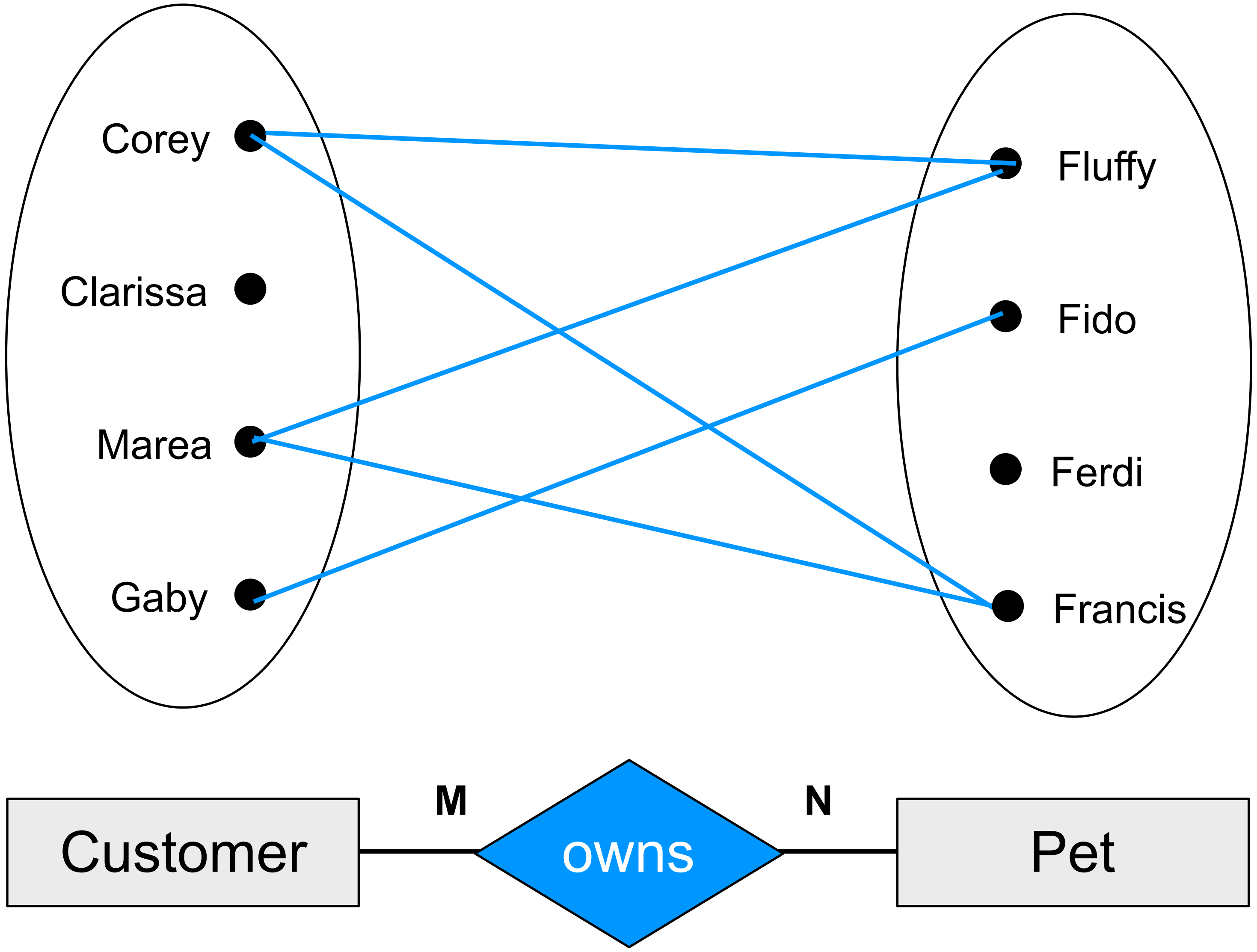
Students are not advised by multiple profs

Professors do not need to advise students (*e.g.*, on leave)



M-to-N Cardinality

- `M-to-N`: Entities in R and S are linked with many entities from each other.



Real-World Assumptions:

Some pets have not been adopted

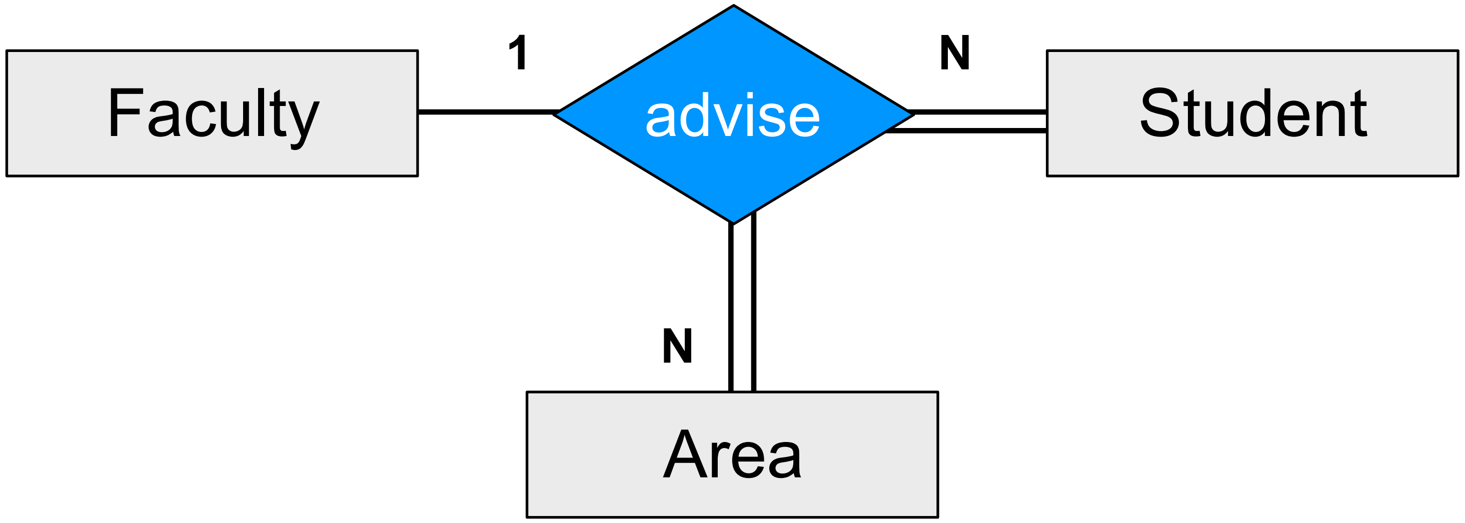
Some customers don't have pets
(*e.g.*, picked up cat food for a friend)

Some pets are owned by multiple customers

Some customers own multiple pets

N-ary Relationship Sets

- We've only seen *binary* relationships, but in fact, relationships can be *n-ary*
- Example of a ternary (involves 3 entities) relationship set



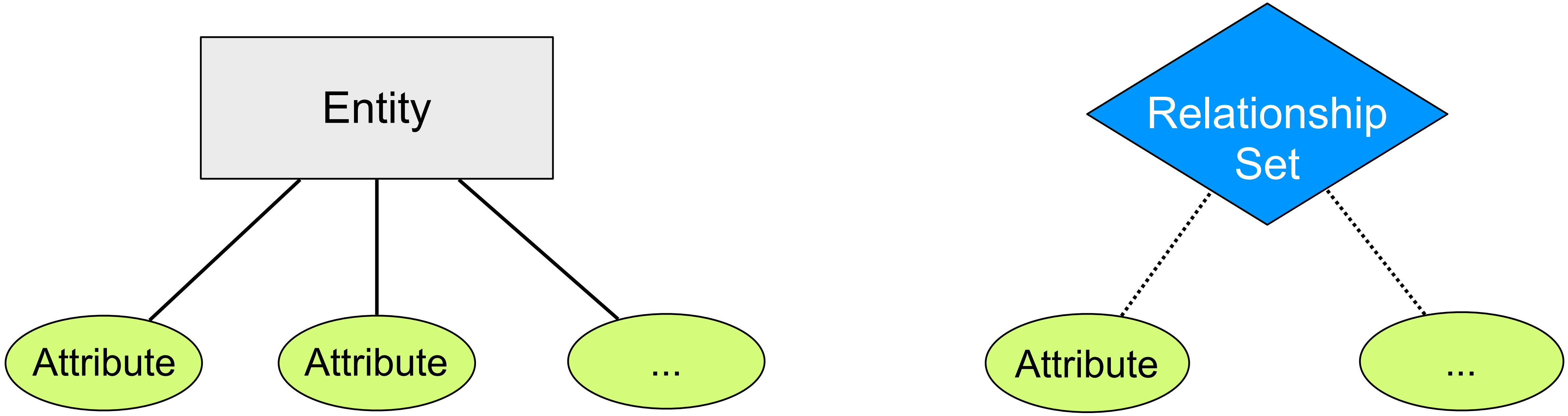
$$advise = \{(\text{Prof David, Fred, Bitmap}), (\text{Prof David, Helena, CS Edu}), (\text{Prof Scott, Eriko, ML}), \ldots\}$$

Topics

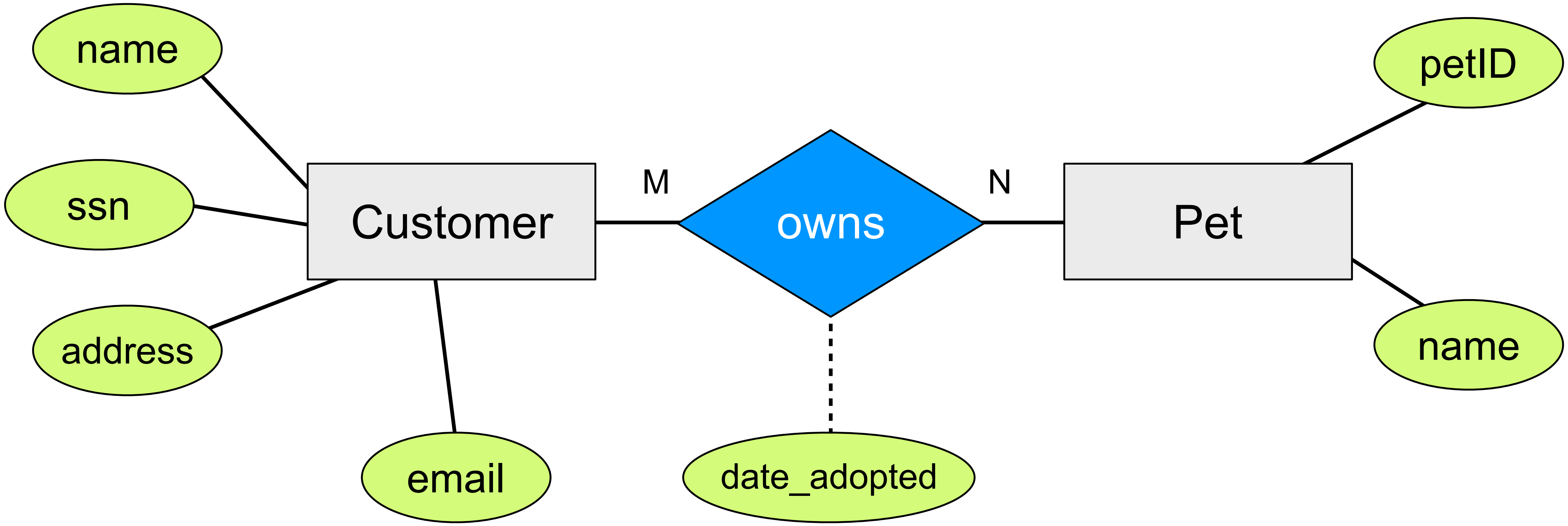
- Motivation
- The ER Building Blocks
  - Entity and Entity Sets
  - Relationship and Relationship Sets
  - Attributes
  - Weak and Strong Entities
- Examples
- ER Reduction to Relations
- Conclusion

Attributes

- Both *entities* and *relationships* can have *attributes*
  - Attributes are the data we wish to store about each entity or relationship set



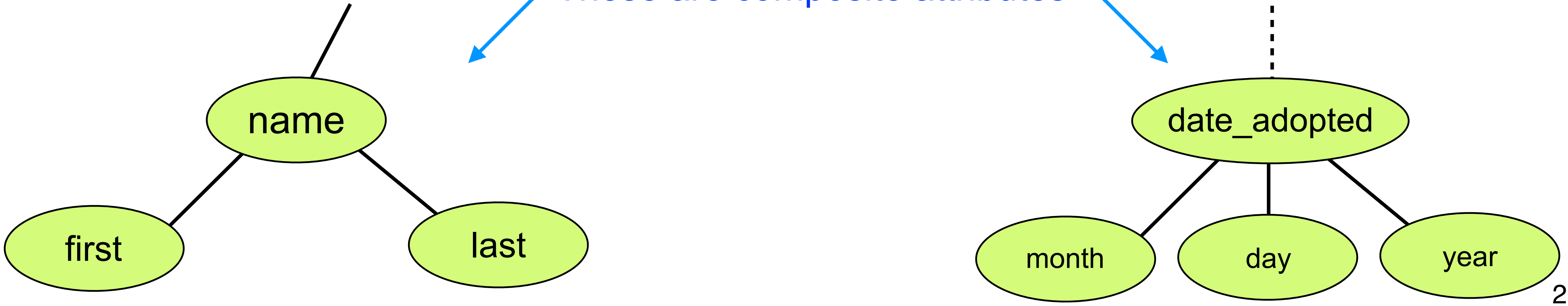
Example


name
ssn
address
email
Customer
M
owns
N
date_adopted
Pet
petID
name

Simple vs. Composite Attributes

- Simple attributes:
  - Atomic, no subparts to them
  - Examples: SSN, an email address
- Composite attributes:
  - Some attributes make sense to be split into subparts
    - Name can be split into first name, middle name, last name
    - Address can be split into street, city, zip

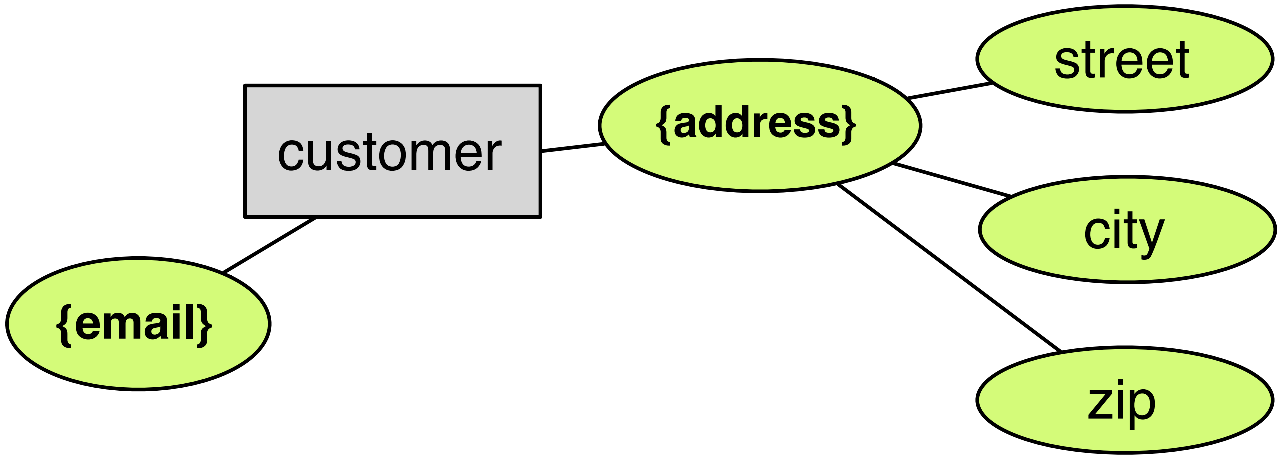
These are composite attributes



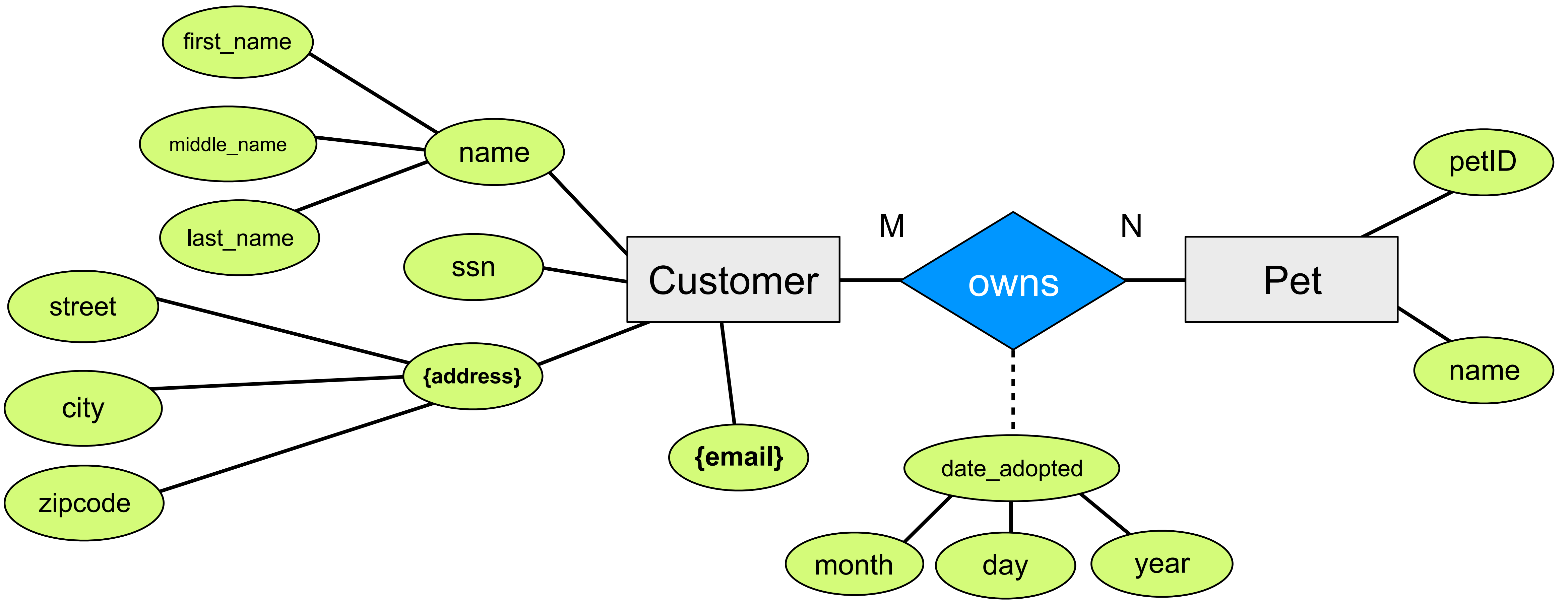
Single-Valued vs. Multivalued Attributes

- Some attributes can take on multiple values
  - A customer can have multiple phone numbers
    - home, work, cell
  - Or multiple emails, or multiple addresses, etc.

- *Multivalued Attributes* are enclosed within {curly braces}

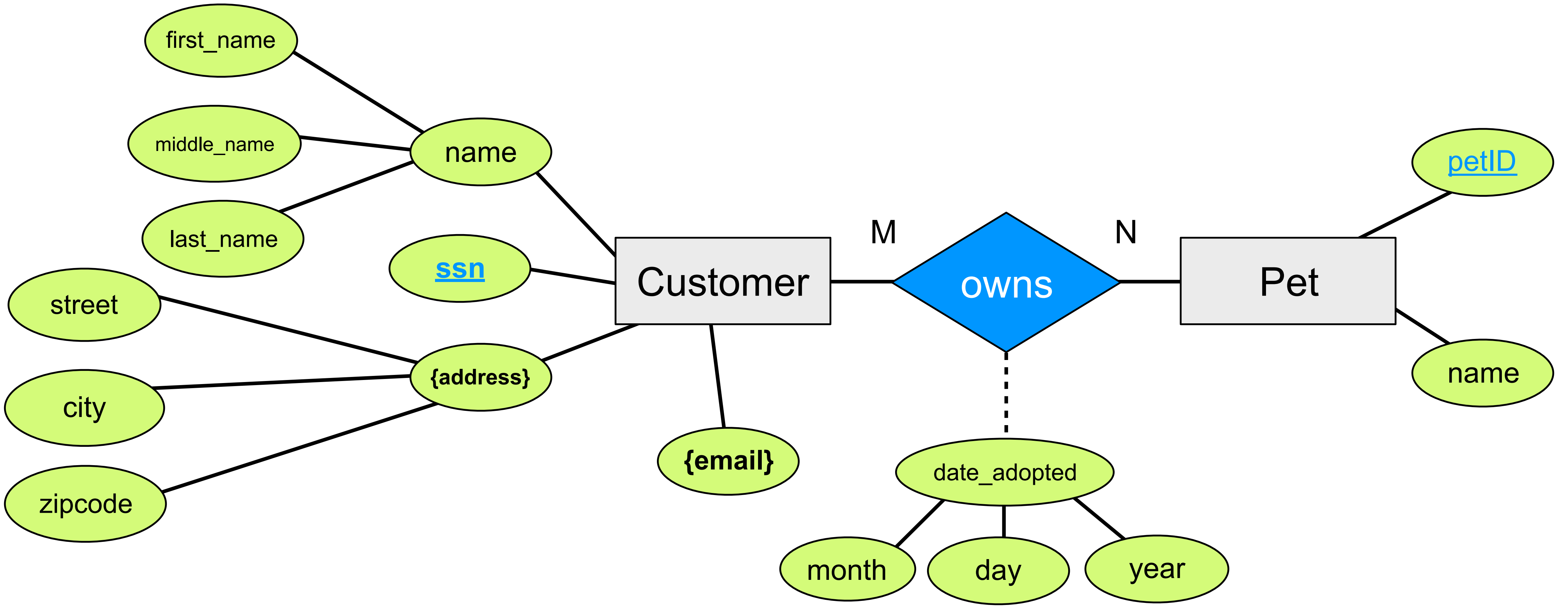


Updated Example



Attributes: Keys for Entities

- *Keys* are the set of attributes that can uniquely identify entities
  - Keys are underlined in the diagram
    - Important: relationship sets need not have a key underlined *(later!)*

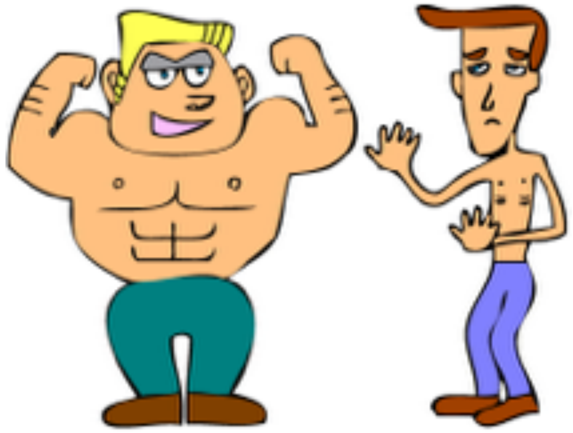


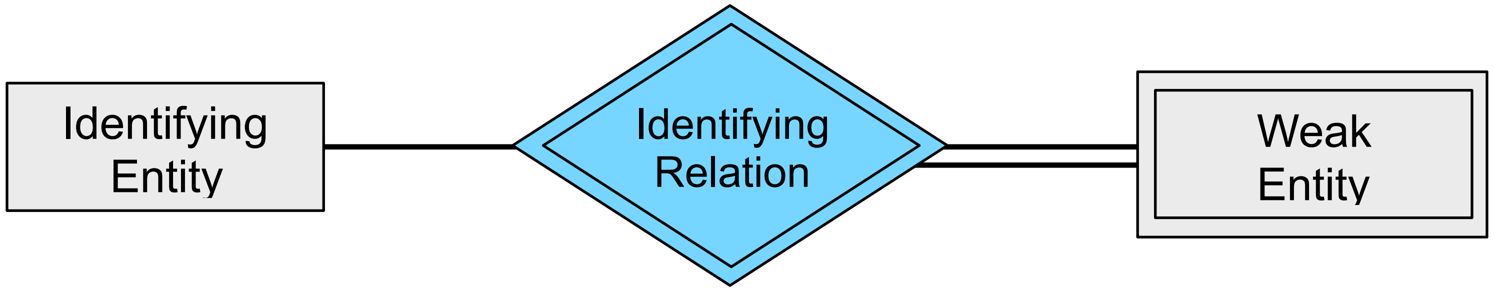
Topics

- Motivation
- The ER Building Blocks
  - Entity and Entity Sets
  - Relationship and Relationship Sets
  - Attributes
  - Weak and Strong Entities
- Examples
- ER Reduction to Relations
- Conclusion

Weak and Strong Entity Sets

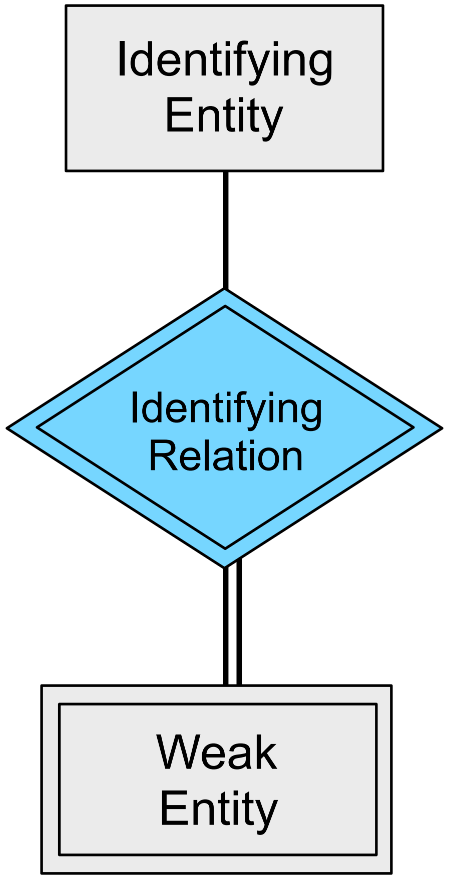
- Entities can be further classified:
  - Strong Entities: What we've been calling "entities"
  - Weak entities exist only due to a strong entity
    - This strong entity is called its *Identifying Entity*
    - Cannot be uniquely identified by its attributes
    - They have a *partial key*, but it's insufficient to uniquely identify any weak entity



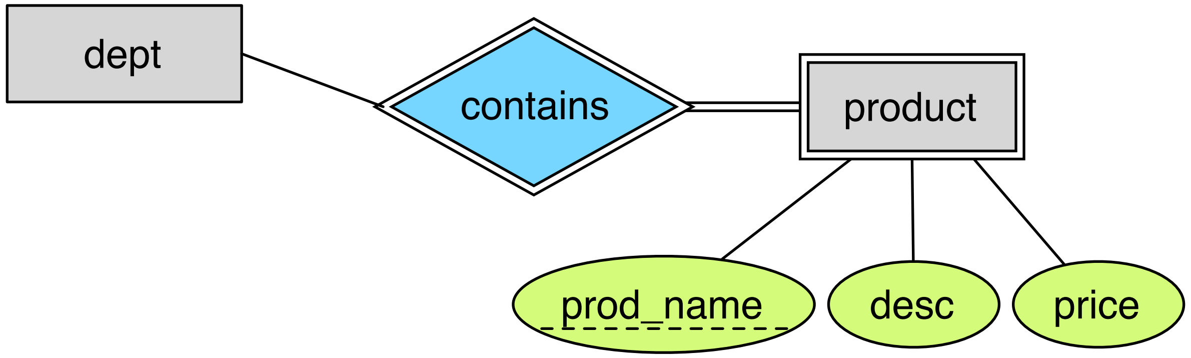


Weak and Strong Entity Sets (Cont.)

- Weak entity is connected to an identifying entity by an *identifying relation*
  - Total participation required *(why?)*



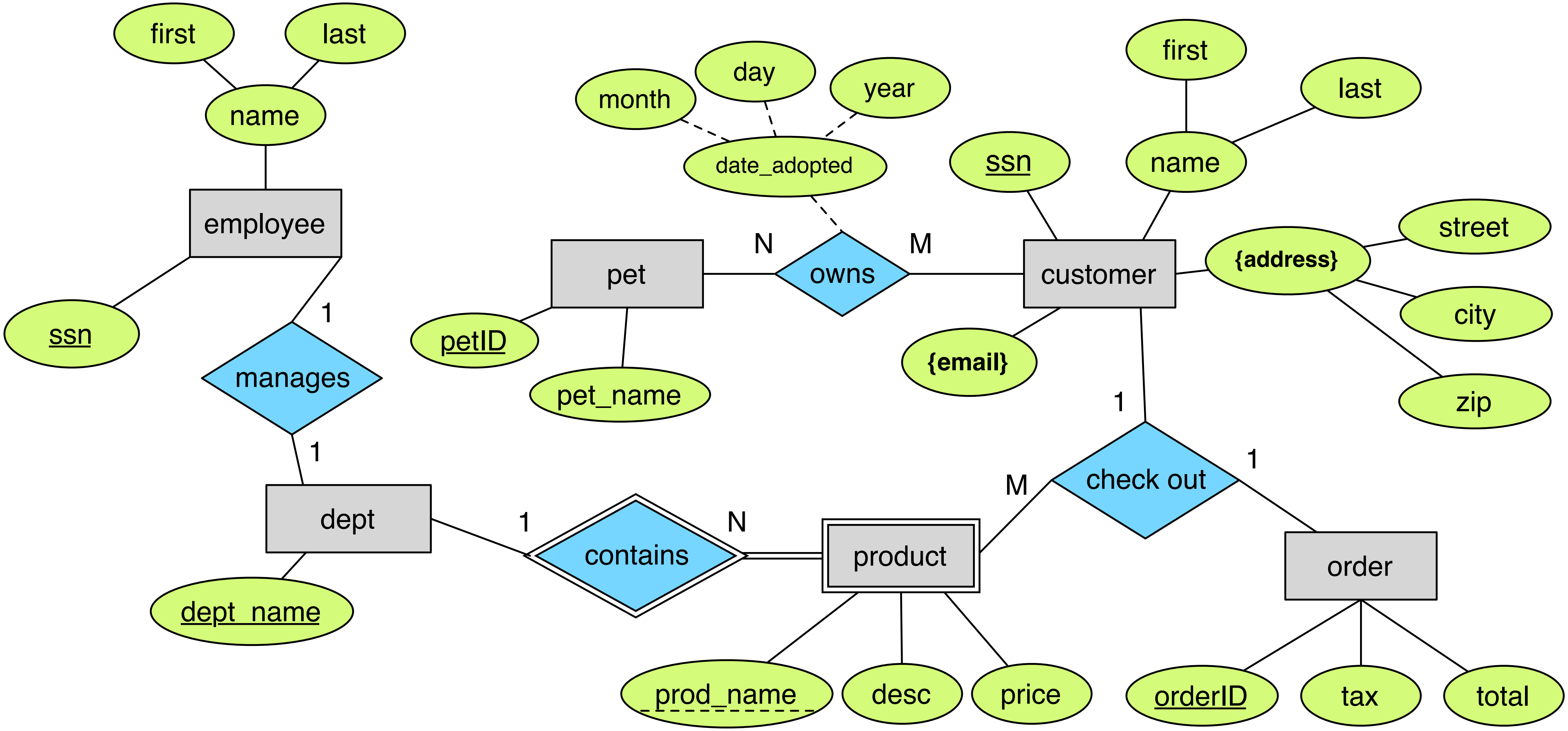
- Examples
  - Employee and their Dependents
  - Course and their Sections
  - Sales Departments and their Products:



Partial keys are not unique

But become unique when taken together with identifying entity's key

Finally...

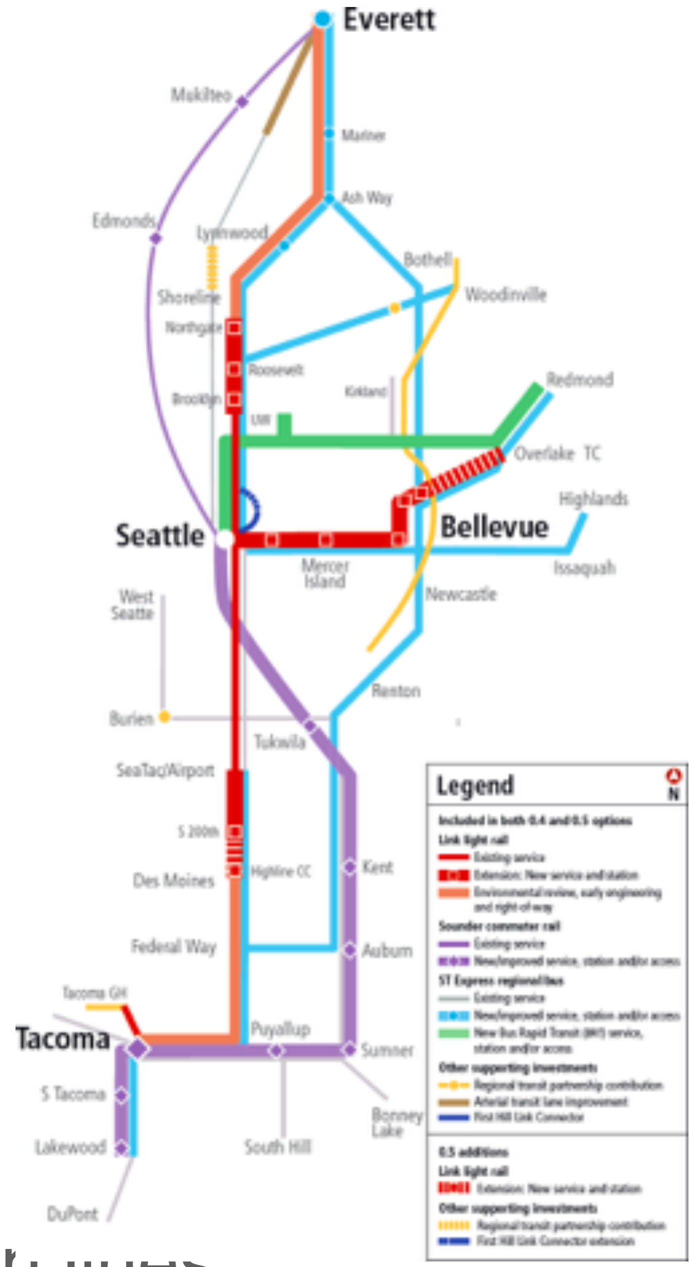

first
last
name
employee
ssn
1
manages
1
dept
dept_name
month
day
year
date_adopted
N
owns
M
pet
petID
pet_name
1
contains
N
product
prod_name
desc
price
first
last
ssn
name
customer
{address}
street
city
zip
{email}
1
M
check out
1
order
orderID
tax
total

Topics

- Motivation
- The ER Building Blocks
  - Entity and Entity Sets
  - Relationship and Relationship Sets
  - Attributes
  - Weak and Strong Entities
- Examples
- ER Reduction to Relations
- Conclusion

Your Turn!

- Model the Sounder Light Rail system with an ER Diagram


Everett
Mukilteo
Mariner
Ash Way
Edmonds
Lynnwood
Bothell
Woodinville
Shoreline
Northgate
Roosevelt
Brooklyn
Kirkland
UW
Redmond
Overlake TC
Highlands
Seattle
Bellevue
Mercer Island
Issaquah
West Seattle
Newcastle
Renton
Burien
Tukwila
SeaTac/Airport
S 200th
Highline CC
Kent
Des Moines
Federal Way
Auburn
Tacoma GH
Tacoma
Puyallup
Sumner
S Tacoma
Bonney Lake
Lakewood
South Hill
DuPont
Legend
N
Included in both 0.4 and 0.5 options
Link light rail
Existing service
Extension: New service and station
Environmental review, early engineering and right-of-way
Sounder commuter rail
Existing service
New/improved service, station and/or access
ST Express regional bus
Existing service
New/improved service, station and/or access
New Bus Rapid Transit (BRT) service, station and/or access
Other supporting investments
Regional transit partnership contribution
Arterial transit lane improvement
First Hill Link Connector
0.5 additions
Link light rail
Extension: New service and station
Other supporting investments
Regional transit partnership contribution
First Hill Link Connector extension

- 4 "lines" (blue, red, green, yellow)
  - Several trains can serve one line
- A line can have many stations
  - Trains can arrive at the same station at various times
- Employees can conduct, or validate tickets on certain lines

Your Turn!

- Model all food carts (not just one business)

Potato Champion

Food Carts

- Assumptions:
  - Carts are mobile and change locations anytime
    - One business can have several carts at various locations
  - Carts have menus, which have menu items and prices
  - Employees can manage, own-carts, or work-for carts

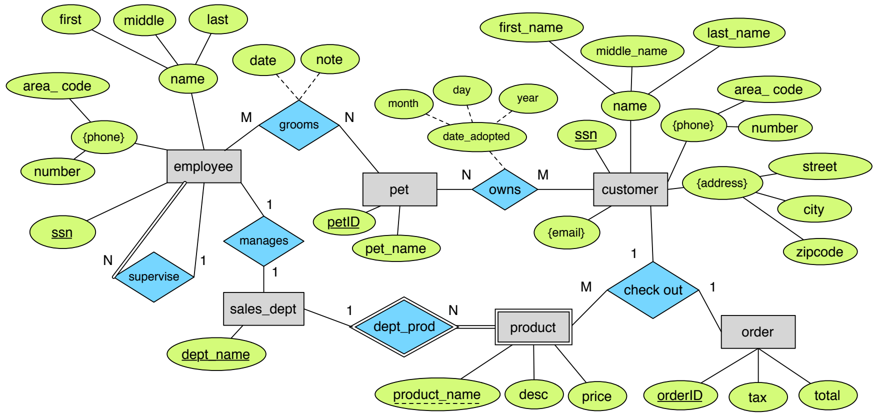
Topics

- Motivation for Entity Relationship Diagrams (ERD)
- The ER Building Blocks
  - Entity and Entity Sets
  - Relationship and Relationship Sets
  - Attributes
  - Weak and Strong Entities
- Examples
- ER Reduction to Relations
- Conclusion

ER-to-Schema Reduction

- An Entity-Relationship model can be converted *(reduced)* to relational schema

- Reduction is a systematic algorithm that inputs an ER model, and outputs a set of relations



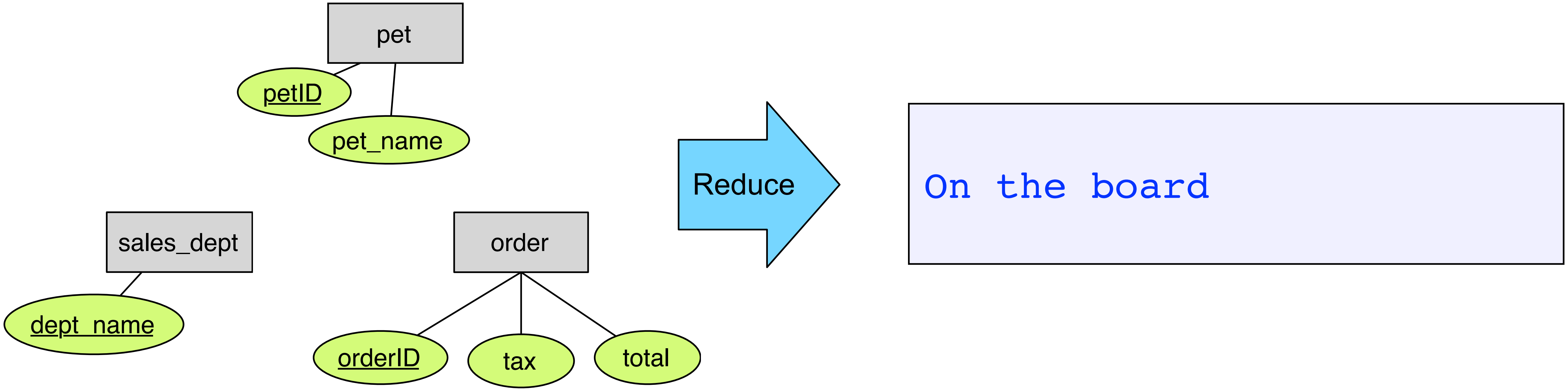
Reduce()

ER-to-Schema Reduction Steps

1. Reduce Strong Entity Sets with Simple Attributes
2. Reduce Strong Entity Sets with Multi-Valued Attributes (MVA)
3. Reduce Weak Entity Sets
4. Reduce Relationship Sets
5. Remove Redundant Schemas
6. Merge Relations

1. Reduce Strong Entity Sets with Only Simple Attributes

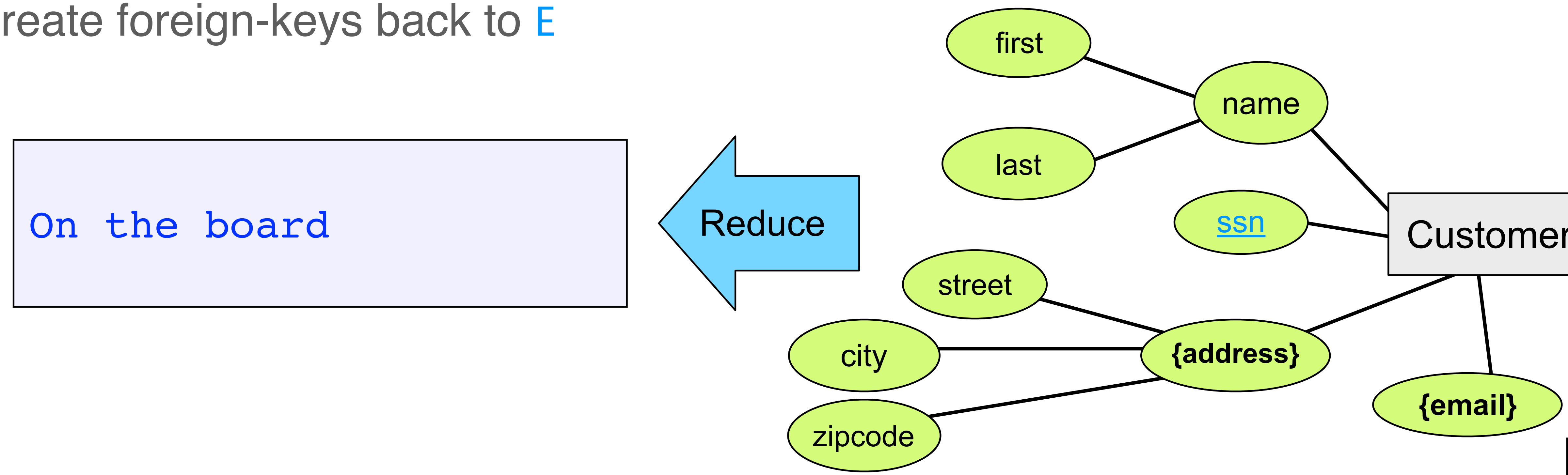
- A *strong entity set* `E` reduces to a schema with the same attributes
- Steps:
  - Create a relation with the entity set name
  - For any *composite* attributes, “flatten them out”
    - Only keep the leaf nodes
  - `key(E)` is the set of underlined attributes



On the board

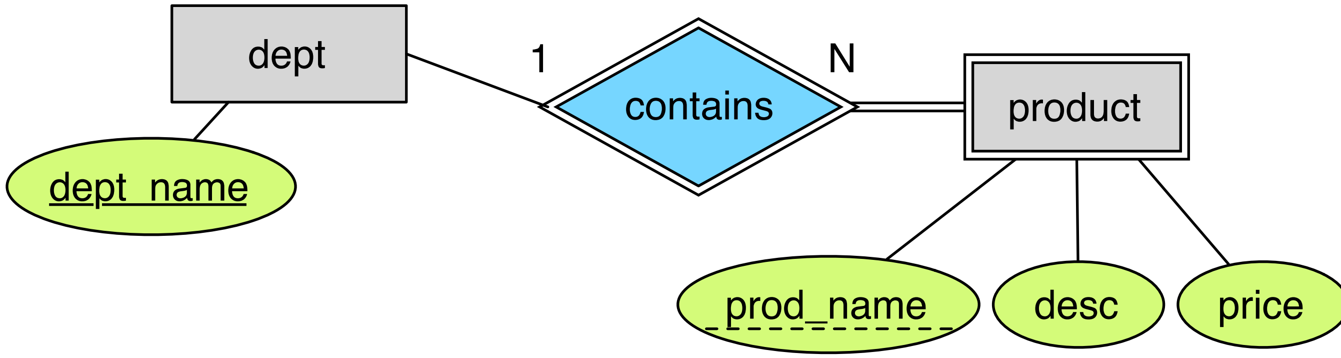
2. Reduce Strong Entity Sets with Multi-Valued Attributes

- Now deal with *strong entity sets with multi-valued attributes (MVA)*
- <u>Steps:</u>
  - Reduce entity set `E` just like before, except the MVAs
  - Each MVA reduces to a separate relation: `EntityNameAttrName`
    - Include `key(E)` as an attribute
    - Set its key to the `key(E)` unioned with all its attributes
    - Create foreign-keys back to `E`



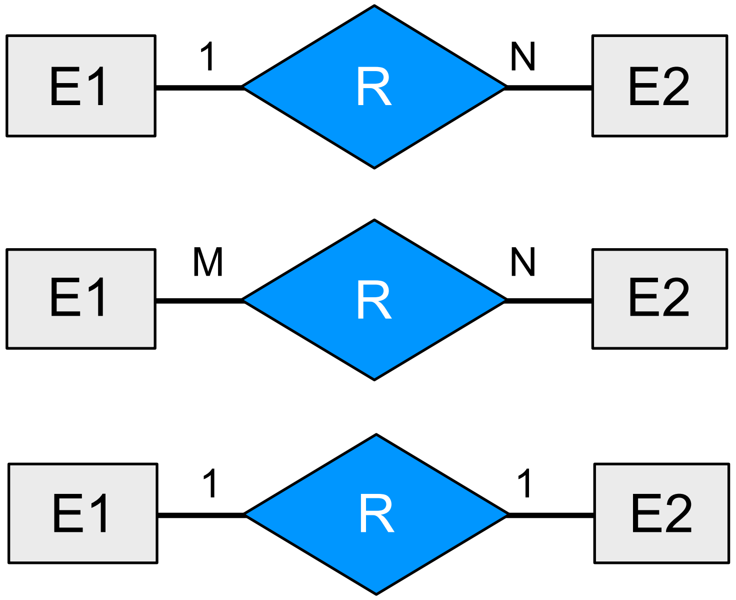
3. Reduce Weak Entity Sets

- Like a strong entity set, a *weak entity set* `W` also becomes a relation with the ER's attributes
  - However, recall we can't identify weak entities by its partial key alone
- Steps:
  - Create relation called `W`
  - Add all attributes
  - Include `key(E')` of the identifying entity, `E'`
  - Set primary key to `{key(E'),partialKey(E)}`
  - Create foreign key from `W`



4. Reduce Relationship Sets

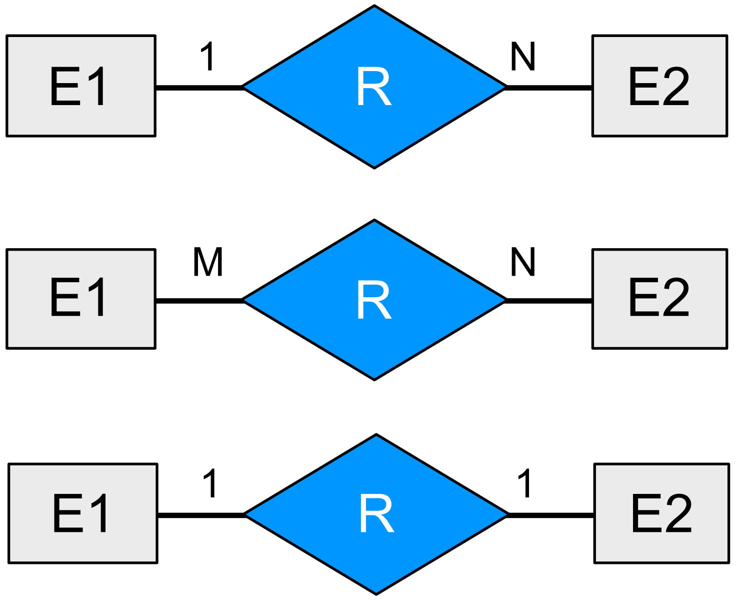
- Steps:
  - For each entity set E participating in the relationship set R, include `key(E)` as attribute
  - Then include any descriptive attributes
  - Set R's key as follows:



  - For each participating entity E, set foreign key from R back to E

4. Reduce Relationship Sets (Cont.)

- <u>Steps:</u>
  - For each entity set E participating in the relationship set R, include key(E) as attribute
  - Then include any descriptive attributes
  - Set R's key as follows:

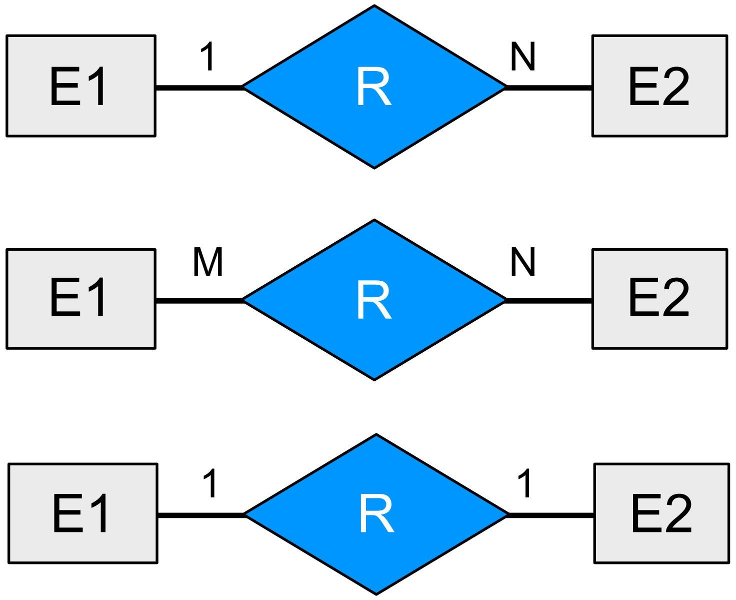


then key(R) = key(E2)

  - For each participating entity E, set foreign key from R back to E

4. Reduce Relationship Sets (Cont.)

- Steps:
  - For each entity set E participating in the relationship set R, include key(E) as attribute
  - Then include any descriptive attributes
  - Set R's key as follows:



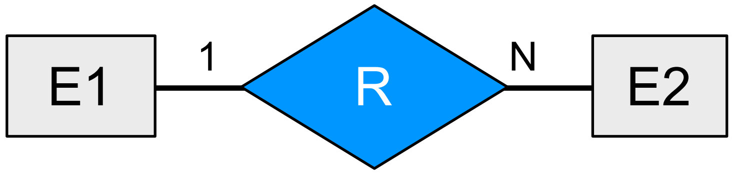
then key(R) = key(E2)

then key(R) = key(E1) ∪ key(E2)

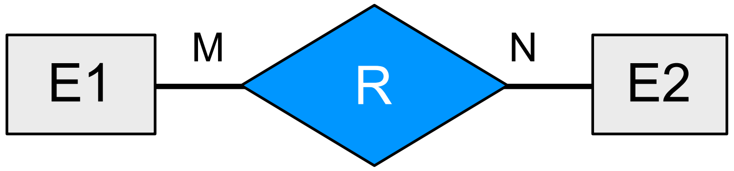
  - For each participating entity E, set foreign key from R back to E

4. Reduce Relationship Sets (Cont.)

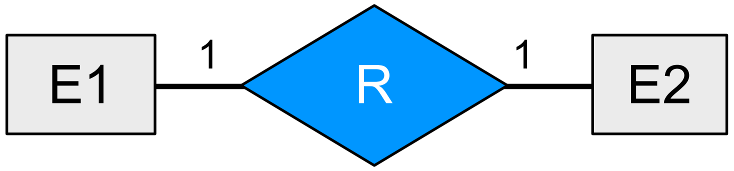
- Steps:
  - For each entity set E participating in the relationship set R, include key(E) as attribute
  - Then include any descriptive attributes
  - Set R's key as follows:



then key(R) = key(E2)



then key(R) = key(E1) ∪ key(E2)



then key(R) = key(E1) *or* key(E2)

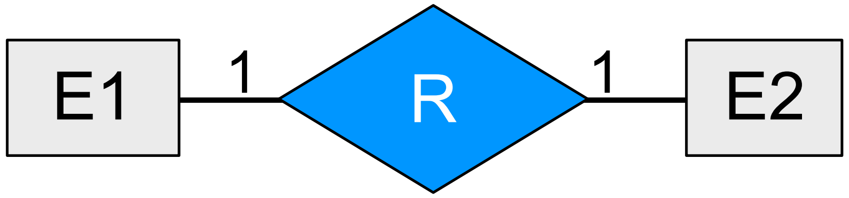
  - For each participating entity E, set foreign key from R back to E

5. Drop Redundant Relations

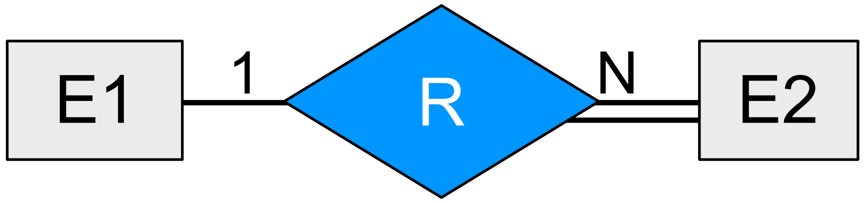
- A simplifying step
- Check your schema diagram for any relations that are redundant
  - Drop duplicate tables, if they exist
  - Entirely possible that there aren't any duplicates to drop

6. Merging Relations

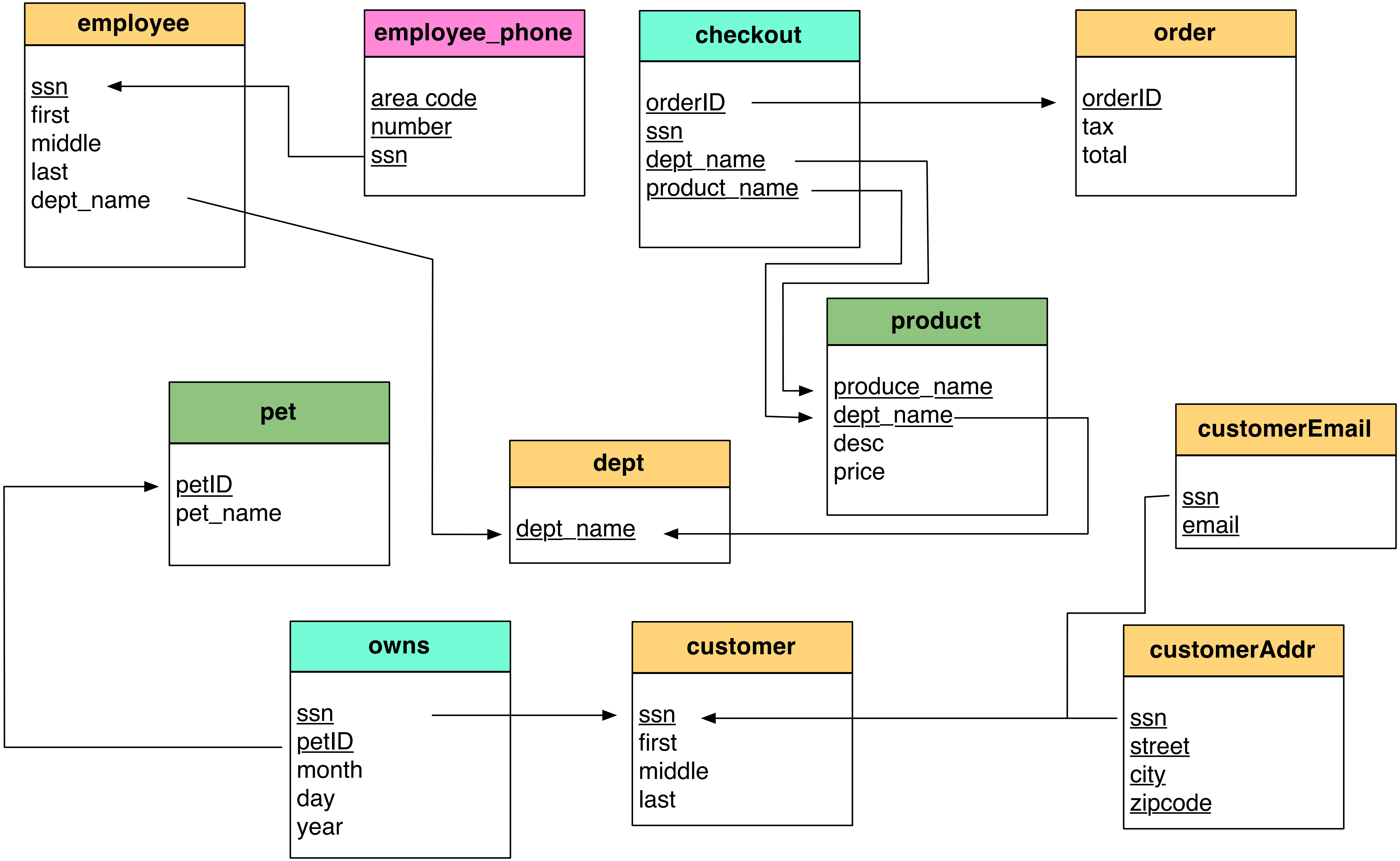
- Merge 1-to-1 relationship sets
  - Choose either E1 and E2 to combine with R into one large relation
  - Create foreign key
  - Drop relation R



- Merge 1-to-N relationship sets
  - For the entity set E2 on the “N” side...
  - If E2 has **total participation**:
    - Combine R into E2
    - Remove duplicate attributes
    - Primary key of merged relation is still *key(E2)*
    - Drop relation R



Ta Da!!! Final Schema



employee
ssn
first
middle
last
dept_name

employee_phone
area code
number
ssn

checkout
orderID
ssn
dept_name
product_name

order
orderID
tax
total

product
produce_name
dept_name
desc
price

pet
petID
pet_name

dept
dept_name

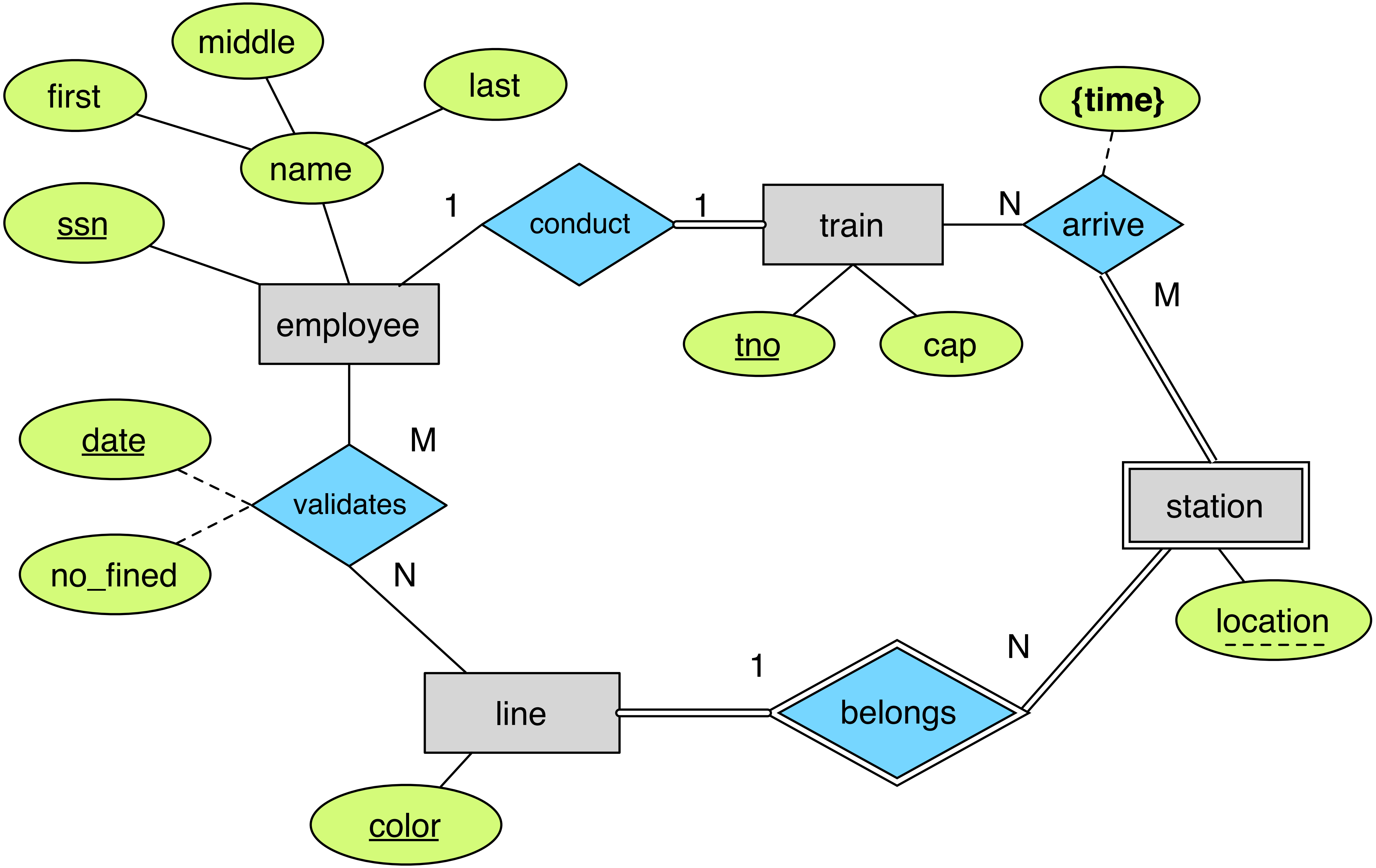
customerEmail
ssn
email

owns
ssn
petID
month
day
year

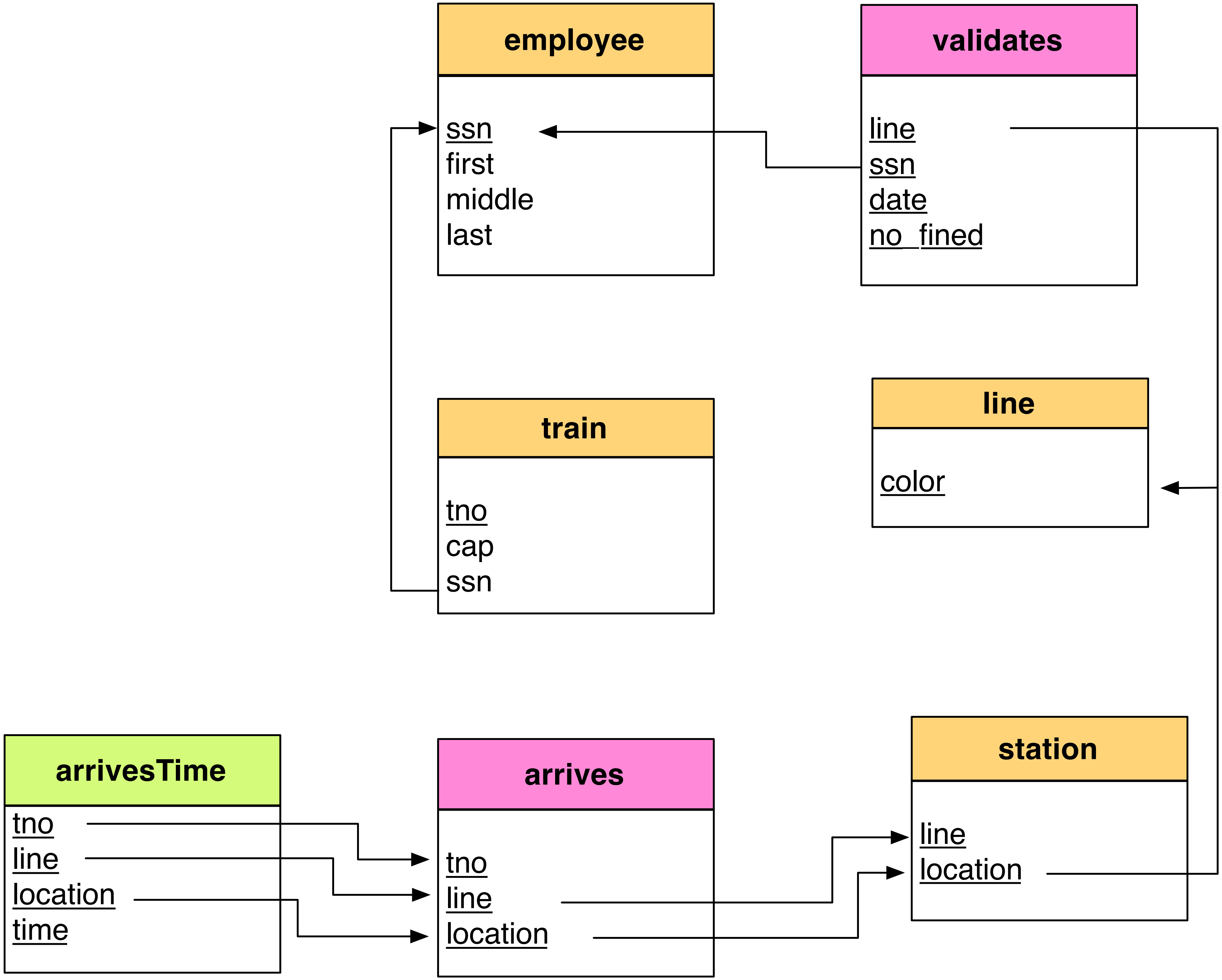
customer
ssn
first
middle
last

customerAddr
ssn
street
city
zipcode

Reduce the Light Rail ERD

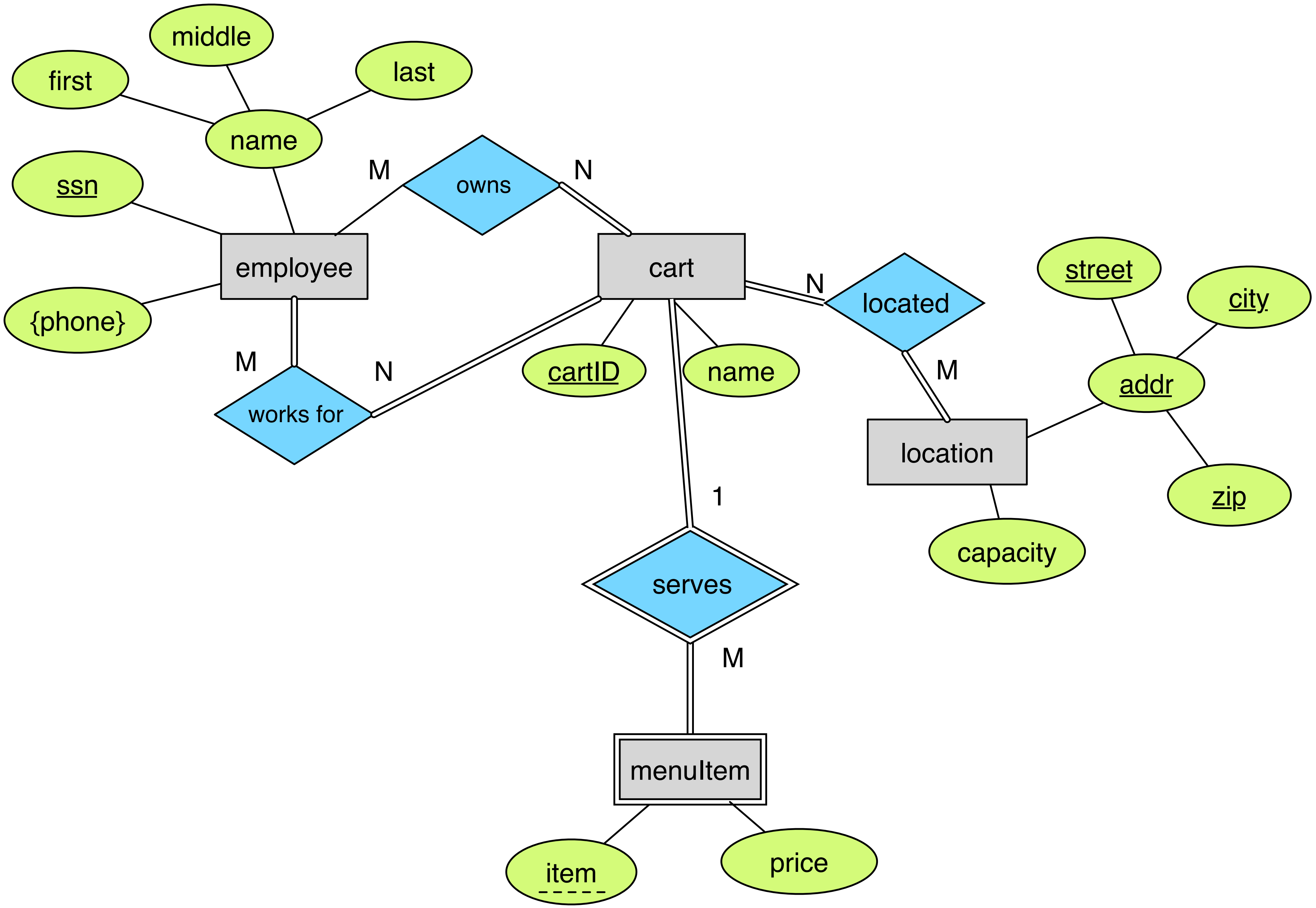


Sounder's Reduced Schema



Your Turn!

- Let's reduce food carts ERD to relation together



Topics

- Motivation
- The ER Building Blocks
  - Entity and Entity Sets
  - Relationship and Relationship Sets
  - Attributes
  - Weak and Strong Entities
- Examples
- ER Reduction to Relations
- Conclusion

In Conclusion...

- Entity Relationship Model is used to communicate and enable understanding of database design

- Basic assumption is database can be described as:
  - A collection of entities, and
  - The relationships among these entities

- Reduction algorithm returns an equivalent relational schema

- But what constitutes good design? *(Next Lecture: Relational Theory)*