Principles of Information and Database Management 198:336 Week 2 – Jan 31

Matthew Stone

Today

Logic and Representation

Entities in IMDB

Relationships in IMDB

Overview of Design Methodology

Function of information manager

Start from a formal language

- Set of sentences
- Each is a symbolic structure with an intended interpretation – as information

Support operations

- TELL a sentence to the IM
  - Give the IM the information that S is true

Function of information manager

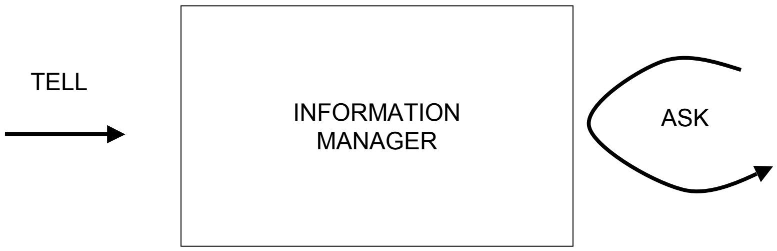
Start from a formal language

- Set of sentences
- Each is a symbolic structure with an intended interpretation – or as an information need

Support operations

- ASK a question to the IM
  - Express an information need to the IM
  - Get back a sentence representing the IM’s answer

Pictorial view



Information Management as Logic

Formal languages are sentences and proofs

IM is a set of logical formulas Σ

TELL : gives the IM a logical formula φ

- The IM gets the information that φ is true

ASK : gives the IM a logical formula φ

- The IM should report its evidence about φ
- Answer is a set of proofs $\Sigma \rightarrow \varphi$

Abstract example

Step 1: **IM = ∅**

Abstract example

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a,c))**

– Give IM the information that the individual represented by **a** stands in the relation represented by **f** to the individual represented by **c.**

Abstract example

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a,c))**

- Give IM the information that the individual represented by **a** stands in the relation represented by **f** to the individual represented by **c.**
- Result: **IM = { f(a,c) }**

Abstract example

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a,c))**

Step 3: **ASK(IM, f(X,c))**

- Ask IM to report proofs that show that some individual **X** stands in the relation represented by **f** to the individual represented by **c.**

Abstract example

Step 1: **IM =** $\emptyset$

Step 2: **TELL(IM, f(a,c))**

Step 3: **ASK(IM, f(X,c))**

- Ask IM to report proofs that show that some individual **X** stands in the relation represented by **f** to the individual represented by **c.**
- IM answers with a single proof of the form

$$\mathbf{f(a,c) \rightarrow f(X,c)\ [X=a]}$$

Abstract example

Step 1: **IM =** $\emptyset$

Step 2: **TELL(IM, f(a,c))**

Step 3: **ASK(IM, f(X,c))**

Step 4: **TELL(IM, f(b,c))**

- Give IM the information that the individual represented by **b** stands in the relation represented by **f** to the individual represented by **c.**

Abstract example

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a,c))**

Step 3: **ASK(IM, f(X,c))**

Step 4: **TELL(IM, f(b,c))**

- – Give IM the information that the individual represented by **b** stands in the relation represented by **f** to the individual represented by **c.**
- – Result: **IM = { f(a,c), f(b,c) }**

Abstract example

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a,c))**

Step 3: **ASK(IM, f(X,c))**

Step 4: **TELL(IM, f(b,c))**

Step 5: **ASK(IM, f(X,c))**

- – Ask IM to report proofs that show that some individual **X** stands in the relation represented by **f** to the individual represented by **c.**

Abstract example

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a))**

Step 3: **ASK(IM, f(X))**

Step 4: **TELL(IM, f(b))**

Step 5: **ASK(IM, f(X))**

- IM answers with two proofs

f(a,c) → f(X,c) [X=a]

f(b,c) → f(X,c) [X=b]

The key idea

Causality + consequence ⇒ correspondence

- **If** you design your system so that you only TELL the IM true facts (causality)
- **And if** your system answers queries using correct logical inference (consequence)
- **Then** all the answers you get from the IM are true (correspondence)

A big idea

Underlies AI

- How can we build a system that acts like it knows stuff?

Underlies psychology

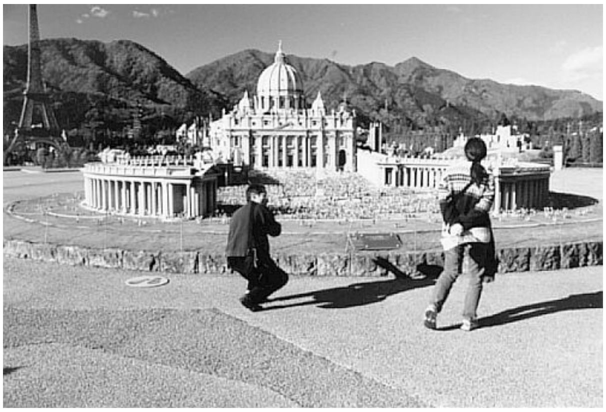
- How can neural events in the brain be related to our ideas about the world?

Shows up in philosophy

- How is it possible, in principle, for sentences or thoughts to be true or false?

Photographs

What’s a photograph about?



It turns out…

This is a photograph of Tobu World Square

- A theme park in Nikko, Japan
- 1/25 scale models of famous buildings
  - Eiffel tower, Great Pyramid, Empire State Building, Forbidden City, Leaning Tower of Pisa, etc.
- Taken by Matt Machlis on vacation
  - Thanks, google!

Here’s an actual picture of St Peter’s Basilica in Vatican City

Googled from Steve Natran

A general fact about representation

A representation is a causal record

- Some entity interacted with the representer
- A new structure was created there
- This structure is patterned after the entity
- If there’s enough detail, something bizarre would have to be going on for the structure to have been created any other way.

Same goes for mental representations

Thought experiment: “Twin Earth”

- Due to philosopher Hilary Putnam

Mental representations

Science fiction scenario

- Imagine another planet exactly like Earth except that in place of H20 they have another substance XYZ.
- You have an exact counterpart over there (a twin!)
- When you are thinking about H20, your twin is thinking about…XYZ.

Same goes for IMDB design

Databases have symbols

- Formalized expressions, such as binary sequences, integers, character strings, or logical atoms.

These are designed to correspond to world

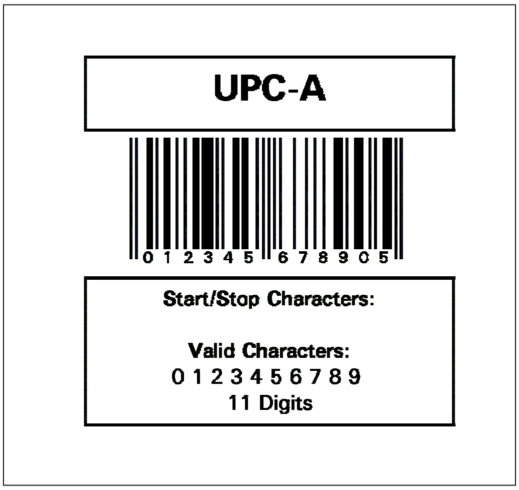
- Entities, properties, relationships, etc.

This works as a representation

- Provided causal connections are there.

Example: bar codes

Products are labeled



Labels are read very reliably

A fixed number of bars per character

- To read code, each bar must be seen.

Most characters are invalid

- If size of one of the bars is misread, you will get an illegal character.

Codes identify product types

Companies are part of UCC

- “Uniform code council”
- UPCs are

  NSC + MFC + PC + CD
- NSC = number system code
- UCC Member companies can get a UPC manufacturer code (MFC)
- Each company makes up its product codes (PC)
- Then there is a check digit

Result

When you scan a product,
it reliably causes system to register a code
that (thus) represents the product type

System’s code 0-71641-81803-3 represents Sanford “Expo” white board cleaner.

Note

We say

- System’s code 0-71641-81803-3 represents Sanford “Expo” white board cleaner.

Symbol: 0-71641-81803-3

corresponds to

Real-world stuff: Sanford “Expo” cleaner

Remember

Meanings ain’t in the head!

- Photograph doesn’t know what it’s a photograph of.
- Twin Earther thinks watery thoughts, but they’re about XYZ.
- Database doesn’t necessarily have information about 0-71641-81803-3.

What does the system know?

System needs explicit information to work with 0-71641-81803-3

- Manufacturer name is “Sanford”
- Product name is “Expo”
- Product type is “cleaner”
- Price is $5.99
- Etc.

Aside

Check out

http://www.upcdatabase.com/

For 56MB of product details.

Normally, manufacturers supply UPC data to trading partneres.

Consequences for IMDB design

Representing entities means:

- Organizing things in the world
- Creating symbols to represent things in DB
- Setting up causal connections so that symbols are used in DB when things are there in the world

Organizing things

Labeling kinds of things

- UPC

Labeling individual objects

- Library books
- EZPass tags
- RFID tags in supply chains (WalMart)

Coding individuals

- Social security numbers

Creating symbols

Store arbitrary numbers!

Creating causal connections

Bar-code scanners

RFID readers

- Get the codes implanted into objects

Security-through-obscurity

- Have people tell you their own numbers
- Passwords, PINs

Biometrics

Automated method of recognizing a person based on a physiological or behavioral characteristic.

- face, fingerprints, hand geometry, handwriting, iris, retinal, vein, and voice.

See http://www.biometrics.org/

Biometrics

Links an event to a particular individual

- a password or token may be used by someone other than the authorized user

Convenient (nothing to carry or remember)

Can provide an audit trail

and is becoming accurate, inexpensive and socially acceptable

See http://www.biometrics.org/

Biometrics, causality & representation

System has internal ID for you:

0-71641-81803-3

Machine knows your physical signature: ✪

When user maches: ✪,
system thinks: 0-71641-81803-3

System's representation 0-71641-81803-3 is really about you.

DB converges with AI?

Robin has internal ID for you:

neuron #0-71641-81803-3

Robin knows your physical signature: ✪

When Robin sees someone matching: ✪,
Robin's neuron #0-71641-81803-3 fires

Robin's neuron #0-71641-81803-3 is a representation that's really about you.

Topic so far: entities

Entity (def)

- A thing with distinct existence, as opposed to a quality or relation.

Thing is more general than entity!

Object is more specific than entity!

- A physical object is something that moves coherently as a unit and maintains its internal structure while in motion.
- An innate human concept? (Elizabeth Spelke)

Topic so far: entities

In DB design:

- each entity you consider has to be distinguishable from the other entities.

Why?

Topic so far: entities

In DB design:

- each entity you consider has to be distinguishable from the other entities.

Why?

- Causality and representation.
- If you can’t tell X and Y apart, you never know when you represent X and when you represent Y.
- Better – work with X and Y’s kind of thing

We can also represent properties and relations

Property (def)

- An attribute, quality or characteristic.

Relation (def)

- What one person or thing has to do with another.

Consequences for IMDB design

Representing properties means:

- Organizing conditions/situations in the world
- Creating symbols for properties in DB
- Setting up causal connections so that the symbol is used in DB in those situations where the property is realized

Bar Code Example

What Property is Represented here?

What Property is Represented here?

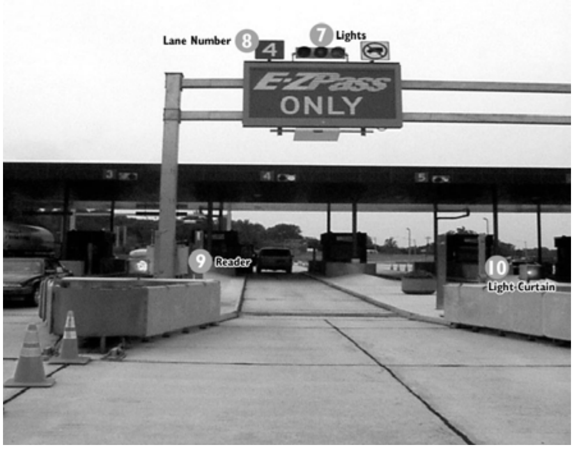
Directly:

- Property of being in visible range of this scanner

Indirectly:

- Property of occurring at a specified position in a physical array of items
  (as orchestrated by a human operator)

EZPass



What Property is Represented here?

What Property is Represented here?

Directly:

- Property of being within range of the reader.

Implicitly:

- Property of driving through a specified toll plaza.

Example, more concrete

Instrument cars and tags so that

- Tag **a** represents Toyota Prius #NJ YY-901
- Tag **b** represents Honda Insight #NJ ZZ-882
- Symbol **c** represents the current time, say Tue Feb 1 11:33:12 EDT 2005
- Symbol **f** represents a relation true of **(v,t)** if vehicle **v** passes through the NJT Exit 9 toll plaza at time **t.**

Example, more concrete

Step 1: **IM = ∅**

Example, more concrete

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a,c))**

- Give IM the information that the individual represented by **a** stands in the relation represented by **f** to the individual represented by **c.**
- **Hey, Toyota Prius #NJ YY-901 just went through the exit 9 toll plaza!**

Example, more concrete

Step 1: **IM =** $\emptyset$

Step 2: **TELL(IM, f(a,c))**

- Give IM the information that the individual represented by **a** stands in the relation represented by **f** to the individual represented by **c.**
- Hey, Toyota Prius #NJ YY-901 just went through the exit 9 toll plaza!
- NB: Meanings ain’t in the head.

Example, more concrete

Step 1: **IM =** $\emptyset$

Step 2: **TELL(IM, f(a,c))**

- Result: **IM = { f(a,c) }**

Example, more concrete

Step 1: **IM = $\emptyset$**

Step 2: **TELL(IM, f(a,c))**

Step 3: **ASK(IM, f(X,c))**

- Ask IM to report proofs that show that some individual **X** stands in the relation represented by **f** to the individual represented by **c.**
- **Hey, what cars went through the exit 9 toll plaza just now?**

Example, more concrete

Step 1: **IM = $\emptyset$**

Step 2: **TELL(IM, f(a,c))**

Step 3: **ASK(IM, f(X,c))**

- Ask IM to report proofs that show that some individual **X** stands in the relation represented by **f** to the individual represented by **c.**
- IM answers with a single proof of the form

$$\mathbf{f(a,c) \rightarrow f(X,c)\ [X=a]}$$

- **It was Toyota Prius #NJ YY-901!**

Example, more concrete

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a,c))**

Step 3: **ASK(IM, f(X,c))**

Step 4: **TELL(IM, f(b,c))**

- – Give IM the information that the individual represented by **b** stands in the relation represented by **f** to the individual represented by **c.**
- – **Hey, Honda Insight NJ ZZ-882 just went through the exit 9 toll plaza!**

Example, more concrete

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a,c))**

Step 3: **ASK(IM, f(X,c))**

Step 4: **TELL(IM, f(b,c))**

- – Result: **IM = { f(a,c), f(b,c) }**

Example, more concrete

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a,c))**

Step 3: **ASK(IM, f(X,c))**

Step 4: **TELL(IM, f(b,c))**

Step 5: **ASK(IM, f(X,c))**

- Ask IM to report proofs that show that some individual **X** stands in the relation represented by **f** to the individual represented by **c.**
- **Hey, what cars went through the exit 9 toll plaza just now?**

Example, more concrete

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a))**

Step 3: **ASK(IM, f(X))**

Step 4: **TELL(IM, f(b))**

Step 5: **ASK(IM, f(X))**

- IM answers with two proofs

f(a,c) → f(X,c) [X=a]

f(b,c) → f(X,c) [X=b]

Example, more concrete

Step 1: **IM = ∅**

Step 2: **TELL(IM, f(a))**

Step 3: **ASK(IM, f(X))**

Step 4: **TELL(IM, f(b))**

Step 5: **ASK(IM, f(X))**

- **It was Toyota Prius #NJ YY-901 and Honda Insight NJ ZZ-882!**

The key idea

Causality + consequence ⇒ correspondence

- **If** you design your system so that you only TELL the IM true facts (causality)
- **And if** your system answers queries using correct logical inference (consequence)
- **Then** all the answers you get from the IM are true (correspondence)

Logic and consequence

Model, M, for first order logic consists of:

D: A set of entities

F: A function mapping each constant symbol to an element of D.

H: A function mapping each a-ary relation symbol to a subset of D^a

Logic and consequence

Value (wrt assignment g, model M):

- If t is a variable, v(t,g) := g(t)
- If t is a constant, v(t,g) := F(t)

Truth (wrt assignment g, model M):

- r(t1,…,tn) is true on g if $\langle$v(t1,g),…,v(tn,g)$\rangle \in$ H(r)
- etc.

Logic and consequence

Truth in model M

- φ is true in M if and only if φ is true in M on any assignment g.

Validity

- φ is valid if and only if it's true in all models

Entailment

- A set of formulas Σ entails φ if and only if φ is true in every model where Σ is true.

Logic and consequence

Inference:

- **Mechanical procedure for constructing proofs – syntactic object deriving conclusion formula φ from assumed formulas Σ**

$$\Sigma \rightarrow \varphi$$

- Don't worry about the details now.

Logic and consequence

Soundness:

If you have a proof $\Sigma \rightarrow \varphi$
then Σ entails φ.

Logic and consequence

Completeness:

If Σ entails φ
then you can find a proof $\Sigma \rightarrow \varphi$.

Why this matters...

Causality induces one specific model.

EZPass example:

- **D: A set of entities, including cars, times**
- **F: A function mapping each constant symbol to an element of D.**
  - **F(a) =** Toyota Prius #NJ YY-901
  - **F(b) =** Honda Insight #NJ ZZ-882
  - **F(c) =** Tue Feb 1 11:33:12 EDT 2005

Why this matters...

Causality provides one specific model.

EZPass example:

- **H:**
  - **H(f) says what cars go through what toll plazas when, as determined by what actually happens in the world.**

Why this matters…

But, meanings ain’t in the head!

From the inside, the system doesn’t see this model.
It only sees the formulas that describe it.

Why this matters…

So, soundness and completeness say what’s needed for the system to act as though it had the information we’ve given.

The key idea

Causality + consequence ⇒ correspondence

- **If** you design your system so that you only TELL the IM true facts (causality)
- **And if** your system answers queries using correct logical inference (consequence)
- **Then** all the answers you get from the IM are true (correspondence)

IMDB design

More about entities and relations:

- The E-R (entity-relationship) model.

Entities in the ER model

Described using a set of attributes

Key

- minimal set of attributes whose values uniquely identify an entity.

Example

Employees of a company: an entity set.

Attributes

- SSN
- Name
- Parking lot

Key:

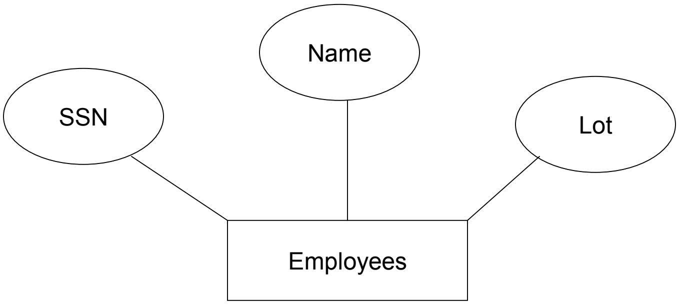
- SSN

Visualization

Entity set: square box.

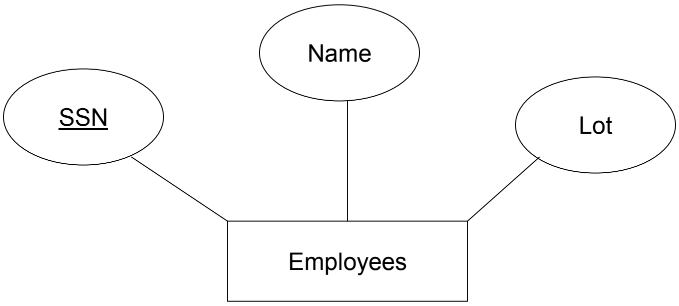
Visualization

Attribute: Oval, connected to entity set.



Visualization

Each attribute in the primary key is underlined.



Relationships

Relationship

- Association among two or more entities
- Like entity, has attributes giving information over and above entities involved called “descriptive attributes”
- Like entity, organized into sets

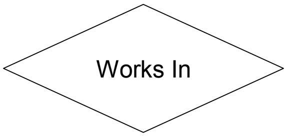
Example

The relationship of an employee working in a department.

- E.g., employee 123-22-3666 works in department 51 (since 1/1/91).
- Attribute “since” says when employee started there.

Visualization

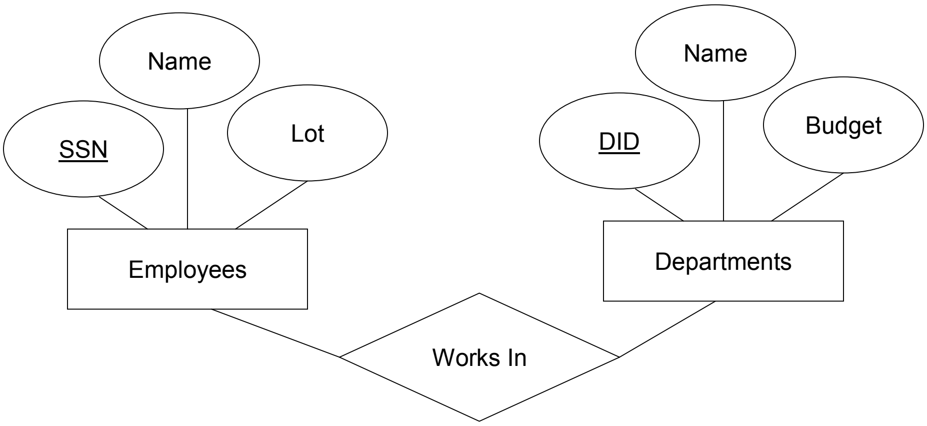
Relationship set is a diamond…



Works In

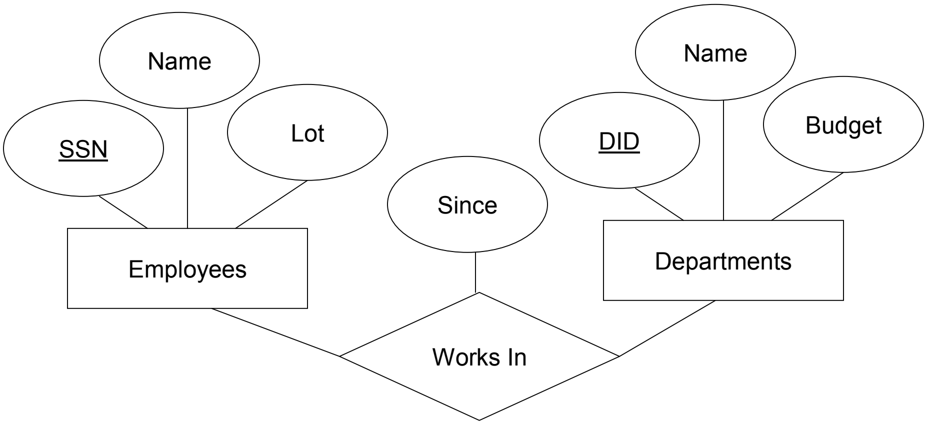
Visualization

…Linked to entity set diagrams…



Visualization

…and descriptive attributes



What’s with this stuff?

Organizing information about the world

- What information is available?
- What information must we keep track of?

ER-models can also answer

- What kinds of relationships are we dealing with?

Characterizing relationship sets

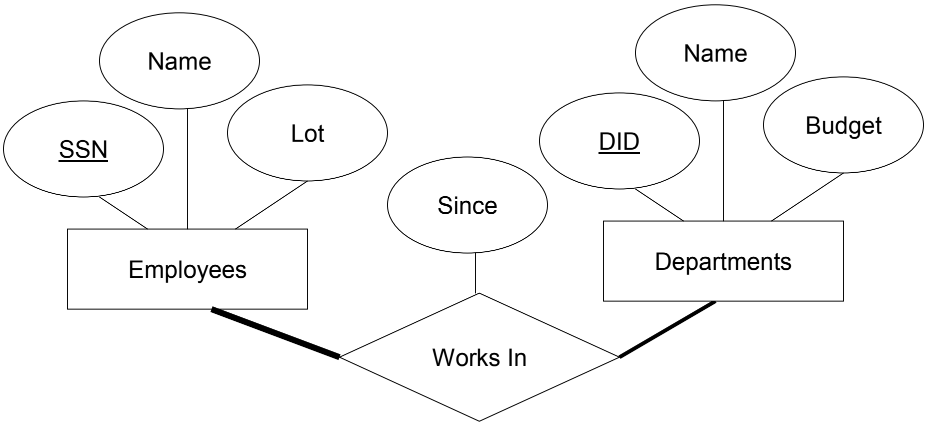
Total participation vs. partial participation

- Every employee works in some department total participation
- Not every employee manages a department partial participation

Visualize total participation by strong connections.

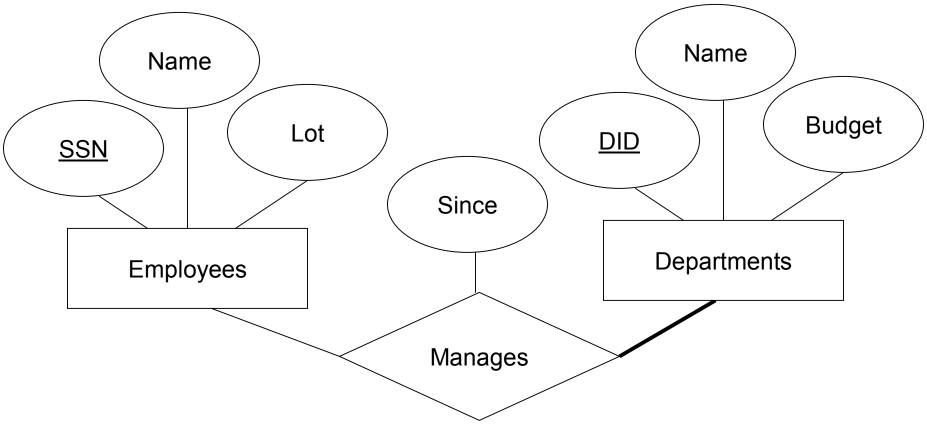
Visualization

Total participation



Visualization

vs. partial participation



Characterizing relationship sets

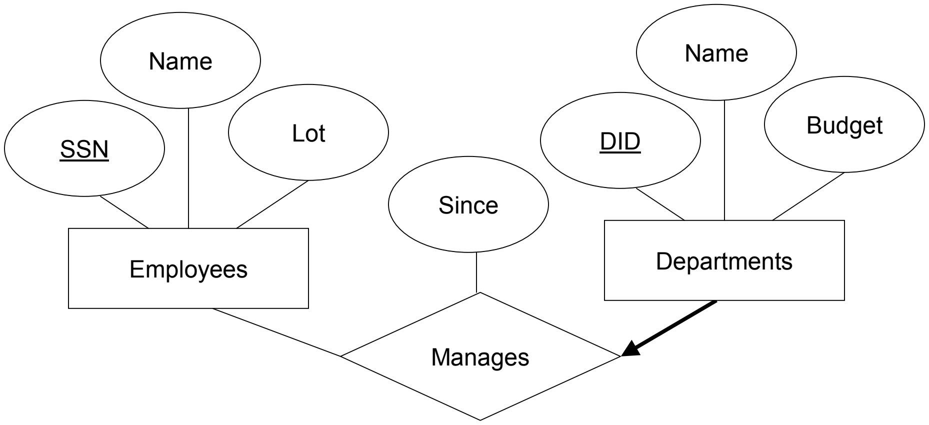
Key constraints and one-to-one relations

Key constraint means that each entity can participate in at most one relationship.

Visualized by an arrow.

Visualization

Departments have at most one manager



Typical design methodology

Requirements analysis

- What must system do?

Conceptual design

- What information is needed?

Logical database design

- What kinds of representations must be involved?

Requirements analysis

Informal work

- Discussions with user groups
- Study of current operating environment
- Analysis of existing applications etc.

Find out

- What must DB do?
- What data should it handle?
- What applications and performance does it need?

Conceptual design

Using ER model

- Develop a high-level description of the data
- Record constraints known to hold over the data

Create a simple description of the data

- Matches how people think of it
- Amenable to implementation

Logical design

Make a database schema

- Choose DBMS
- Translate conceptual design into data model

Typical design methodology

Schema refinement

- Normalizing relations and other streamlining operations

Physical DB design

- Make sure DB meets performance criteria, perhaps retuning schemas

Application and security design

- Human factors and system integration issues