CS 235: Introduction to Databases

Svetlozar Nestorov

Lecture Notes #4

The Big Picture

- Stages of building a database application:
- Real-world domain.
  - understand client needs.
- Design data model:
  - using entity-relationship (E/R) model
- Database data model:
  - using relational model
- Create schema in DBMS, load data.
- Open for business!

Outline

- Relational model.
- From E/R diagrams to relations.

Relational Model

- Table = *relation*.
- Column headers = *attributes*.
- Row = *tuple.*
- *Beers* example:

name	*manf*
Honkers Ale	Goose Island
BudLite	A.B.
…	…

Relational Model

- Relation schema:
  - name (attributes)
  - other structure info., e.g., keys, other constraints.
- Example: *Beers(name, manf)*.
- Order of attributes is arbitrary.
  - In practice we need to assume the order given in the relation schema.
- Relation instance is current set of rows for a relation schema.
- Database schema is collection of relation schemas.
- "A Relational Model of Data for Large Shared Data Banks" by E. F. Codd in *Communications of ACM, Vol 13. No. 6, June 1970*

Why Relations?

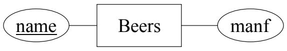
- Very simple model.
- *Often* a good match for the way we think about our data.
- Abstract model that underlies SQL, the most important language in DBMS's today.
  - But SQL uses *bags* while the abstract relational model is set-oriented.

Relational Design

- Simplest approach (not always best):
  - – convert each E.S. to a relation
  - – convert each relationship to a relation.

Entity Set → Relation

- E.S. attributes become relational attributes.



- Becomes:

  Beers(name, manf)

Keys in Relations

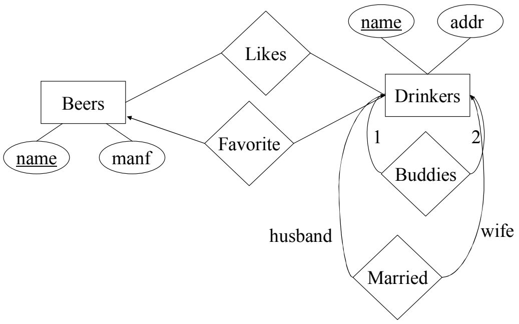
- An attribute or set of attributes K is a key for a relation *R* if we expect that in no instance of *R* will two different tuples agree on all the attributes of *K*.
- Indicate a key by underlining the key attributes.
- Example: If *name* is a key for *Beers*:

  Beers(name, manf)

E/R Relationships → Relations

- Relation has attribute(s) for *key* attributes of each E.S. that participates in the relationship.
- Add any attributes that belong to the relationship itself.
- Renaming attributes OK.
  - – Essential if multiple roles for an E.S.

Example



Combining Relations

- Common case: Relation for an E.S. *E* plus the relation for some *many-one relationship* from *E* to another E.S
- Example:
  - – Combine *Drinkers(name, addr)* with *Favorite(drinker, beer).*
  - – Resulting in: *Drinkers1(name,addr,favBeer).*
- Danger in pushing this idea too far: redundancy.
- Example:
  - – Combining *Drinker* swith *Likes* causes the drinker's address to be repeated, viz.:

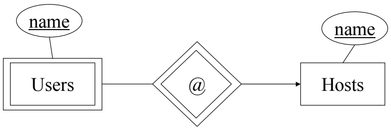
name	*addr*	*beer*
Mike	111 E Ohio	Guinness
Mike	111 E Ohio	Newcastle

- The difference: *Favorite* is many-one; *Likes* is many-many.

Weak Entity Sets, Relationships → Relations

- Relation for a weak E.S. must include its full key (i.e., attributes of related entity sets) as well as its own attributes.
- A supporting (double-diamond) relationship yields a relation that is actually redundant and should be deleted from the database schema.

Example



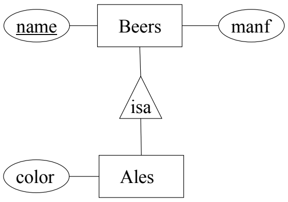
Example

Hosts(name)
Users(name, hostName)
At(userName, hostName, hostName2)

- In *At, hostName* and *hostName2* must be the same host, so delete one of them.
- Then, *Users* and *At* become the same relation; delete one of them.
- In this case, *Hosts*' schema is a subset of *Users*' schema. Delete *Hosts*?

Subclasses → Relations

- Three approaches:
  - Object-oriented
  - E/R style
  - Using nulls



Object-oriented Style

- Each entity is in one class.
- Create a relation for each class, with all the attributes for that class.
  - Don't forget inherited attributes.

name	*manf*
BudLite	A.B.

Beers

name	*manf*	*color*
Honkers Ale	Goose Island	dark

Ales

E/R Style

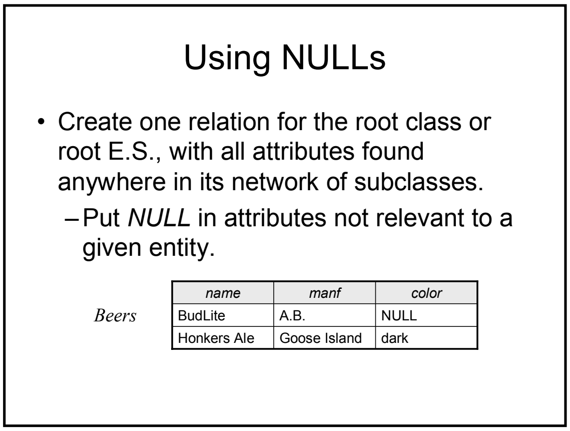
- An entity is in a network of classes related by *isa*.
- Create one relation for each E.S.
  - Relation has only the attributes attached to that E.S. + key.

name	*manf*
BudLite	A.B.
Honkers Ale	Goose Island

Beers

name	*color*
Honkers Ale	dark

Ales

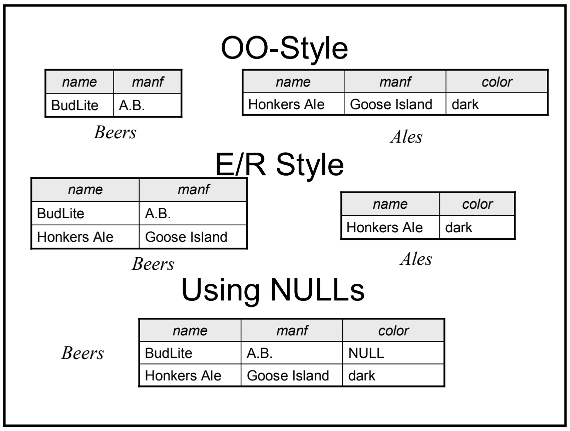


Using NULLs

- Create one relation for the root class or root E.S., with all attributes found anywhere in its network of subclasses.
  - Put *NULL* in attributes not relevant to a given entity.

Beers

name	*manf*	*color*
BudLite	A.B.	NULL
Honkers Ale	Goose Island	dark



OO-Style

name	*manf*
BudLite	A.B.

Beers

name	*manf*	*color*
Honkers Ale	Goose Island	dark

Ales

E/R Style

name	*manf*
BudLite	A.B.
Honkers Ale	Goose Island

Beers

name	*color*
Honkers Ale	dark

Ales

Using NULLs

Beers

name	*manf*	*color*
BudLite	A.B.	NULL
Honkers Ale	Goose Island	dark