CSE 344

FEBRUARY 28TH – ENTITIES
ADMINISTRIVIA

• HW7/8 out
  • Make sure that you’re tagging assignments properly
  • For HW8, only first tag will be graded
• Remember: 5 late days per person
  • Accurate through HW5 on canvas
• OQ #6 out tonight, OQ #7 next week
DATABASE DESIGN

• **What it is:**
  • Starting from scratch, design the database schema: relation, attributes, keys, foreign keys, constraints etc

• **Why it’s hard**
  • The database will be in operation for a very long time (years). Updating the schema while in production is very expensive (why?)
DATABASE DESIGN

Consider issues such as:

- What entities to model
- How entities are related
- What constraints exist in the domain

Several formalisms exists

- We discuss E/R diagrams
- UML, model-driven architecture

Reading: Sec. 4.1-4.6
DATABASE DESIGN PROCESS

Conceptual Model:

Relational Model:
Tables + constraints
And also functional dep.

Normalization:
Eliminates anomalies

Conceptual Schema

Physical storage details
Physical Schema
ENTITY / RELATIONSHIP DIAGRAMS

Entity set = a class
  • An entity = an object

Attribute

Relationship

Product

city

makes
Every entity set must have a key

**Product**

- **name**
- **price**
WHAT IS A RELATION?

A mathematical definition:
- if A, B are sets, then a relation R is a subset of A × B

A={1,2,3}, B={a,b,c,d},
A × B = {(1,a), (1,b), . . . , (3,d)}
R = {(1,a), (1,c), (3,b)}

makes is a subset of Product × Company:
MULTIPLICITY OF E/R RELATIONS

one-one:

- 1
- 2
- 3
- a
- b
- c
- d

many-one

- 1
- 2
- 3
- a
- b
- c
- d

many-many

- 1
- 2
- 3
- a
- b
- c
- d
What does this say?
What does this say?
How do we model a purchase relationship between buyers, products and stores?

Can still model as a mathematical set (How?)

As a set of triples \( \subseteq \text{Person} \times \text{Product} \times \text{Store} \)
Q: What does the arrow mean?

A: Any person buys a given product from at most one store.

[Fine print: Arrow pointing to E means that if we select one entity from each of the other entity sets in the relationship, those entities are related to at most one entity in E]
Q: What does the arrow mean?

A: Any person buys a given product from at most one store AND every store sells to every person at most one product.
CONVERTING MULTI-WAY RELATIONSHIPS TO BINARY

Arrows go in which direction?
CONVERTING MULTI-WAY RELATIONSHIPS TO BINARY

Make sure you understand why!
3. DESIGN PRINCIPLES

What’s wrong?

Moral: Be faithful to the specifications of the application!
DESIGN PRINCIPLES: WHAT’S WRONG?

Moral: pick the right kind of entities.
Moral: don’t complicate life more than it already is.
 ENTITY SET TO RELATION

### Product

<table>
<thead>
<tr>
<th>prod-ID</th>
<th>category</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo55</td>
<td>Camera</td>
<td>99.99</td>
</tr>
<tr>
<td>Pokemn19</td>
<td>Toy</td>
<td>29.99</td>
</tr>
</tbody>
</table>
Represent this in relations
N-N RELATIONSHIPS TO RELATIONS

Orders\( (\text{prod-ID}, \text{cust-ID}, \text{date}) \)

Shipment\( (\text{prod-ID}, \text{cust-ID}, \text{name}, \text{date}) \)

Shipping-Co\( (\text{name}, \text{address}) \)

<table>
<thead>
<tr>
<th>prod-ID</th>
<th>cust-ID</th>
<th>name</th>
<th>date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo55</td>
<td>Joe12</td>
<td>UPS</td>
<td>4/10/2011</td>
</tr>
<tr>
<td>Gizmo55</td>
<td>Joe12</td>
<td>FEDEX</td>
<td>4/9/2011</td>
</tr>
</tbody>
</table>
N-1 RELATIONSHIPS TO RELATIONS

Represent this in relations
N-1 RELATIONSHIPS TO RELATIONS

Orders \((\text{prod-ID, cust-ID, date1, name, date2})\)

Shipping-Co \((\text{name, address})\)

Remember: no separate relations for many-one relationship
MULTI-WAY RELATIONSHIPS TO RELATIONS

Try this at home!

Purchase \((\text{prod-ID, ssn, name})\)
Some objects in a class may be special
• define a new class
• better: define a *subclass*

So --- we define subclasses in E/R
MODELING SUBCLASSES

Product

- name
- category
- price
- isa
- isa

Software Product
- platforms

Educational Product
- Age Group
Other ways to convert are possible
Say: each piece of furniture is owned either by a person or by a company
Say: each piece of furniture is owned either by a person or by a company

Solution 1. Acceptable but imperfect (What’s wrong?)
MODELING UNION TYPES WITH SUBCLASSSES

Solution 2: better, more laborious

- isa FurniturePiece
  - ownedBy Owner
  - isa Person
  - isa Company
WEAK ENTITY SETS

Entity sets are weak when their key comes from other classes to which they are related.

Team(sport, number, universityName)
University(name)
WHAT ARE THE KEYS OF R?
An integrity constraint is a condition specified on a database schema that restricts the data that can be stored in an instance of the database.

ICs help prevent entry of incorrect information

How? DBMS enforces integrity constraints

- Allows only legal database instances (i.e., those that satisfy all constraints) to exist
- Ensures that all necessary checks are always performed and avoids duplicating the verification logic in each application
Finding constraints is part of the modeling process.
Commonly used constraints:

**Keys:** social security number uniquely identifies a person.

**Single-value constraints:** a person can have only one father.

**Referential integrity constraints:** if you work for a company, it must exist in the database.

**Other constraints:** peoples’ ages are between 0 and 150.
No formal way to specify multiple keys in E/R diagrams
SINGLE VALUE CONSTRAINTS

makes vs. makes
Each product made by at most one company. Some products made by no company.

Each product made by exactly one company.
Q: What does this mean?
A: A Company entity cannot be connected by relationship to more than 99 Product entities.
Constraints in SQL:

- Keys, foreign keys
- Attribute-level constraints
- Tuple-level constraints
- Global constraints: assertions

The more complex the constraint, the harder it is to check and to enforce.
KEY CONSTRAINTS

Product(name, category)

CREATE TABLE Product (name CHAR(30) PRIMARY KEY, category VARCHAR(20))

OR:

CREATE TABLE Product (name CHAR(30), category VARCHAR(20), PRIMARY KEY (name))
KEYS WITH MULTIPLE ATTRIBUTES

CREATE TABLE Product ( 
    name CHAR(30),
    category VARCHAR(20),
    price INT,
    PRIMARY KEY (name, category))

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Gadget</td>
<td>10</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
<td>20</td>
</tr>
<tr>
<td>Gizmo</td>
<td>Photo</td>
<td>30</td>
</tr>
<tr>
<td>Gizmo</td>
<td>Gadget</td>
<td>40</td>
</tr>
</tbody>
</table>
CREATE TABLE Product (
    productID CHAR(10),
    name CHAR(30),
    category VARCHAR(20),
    price INT,
    PRIMARY KEY (productID),
    UNIQUE (name, category))

There is at most one PRIMARY KEY; there can be many UNIQUE
FOREIGN KEY CONSTRAINTS

CREATE TABLE Purchase (  
    prodName CHAR(30)  
    REFERENCES Product(name),  
    date DATETIME)

prodName is a foreign key to Product(name)  
name must be a key in Product

Referential integrity constraints

May write just Product if name is PK
FOREIGN KEY CONSTRAINTS

Example with multi-attribute primary key

```sql
CREATE TABLE Purchase (  
  prodName CHAR(30),  
  category VARCHAR(20),  
  date DATETIME,  
  FOREIGN KEY (prodName, category)  
  REFERENCES Product(name, category)
)
```

(name, category) must be a KEY in Product
WHAT HAPPENS WHEN DATA CHANGES?

Types of updates:
In Purchase: insert/update
In Product: delete/update

<table>
<thead>
<tr>
<th>Product</th>
<th>Purchase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Category</td>
</tr>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>
WHAT HAPPENS WHEN DATA CHANGES?

SQL has three policies for maintaining referential integrity:

- **NO ACTION** reject violating modifications (default)
- **CASCADE** after delete/update do delete/update
- **SET NULL** set foreign-key field to NULL
- **SET DEFAULT** set foreign-key field to default value

  - need to be declared with column, e.g.,
    CREATE TABLE Product (pid INT DEFAULT 42)
MAINTAINING REFERENTIAL INTEGRITY

CREATE TABLE Purchase (  
    prodName CHAR(30),  
    category VARCHAR(20),  
    date DATETIME,  
    FOREIGN KEY (prodName, category)  
        REFERENCES Product(name, category)  
    ON UPDATE CASCADE  
    ON DELETE SET NULL  
)
CONSTRAINTS ON ATTRIBUTES AND TUPLES

Constraints on attributes:
- **NOT NULL** -- obvious meaning...
- **CHECK** condition -- any condition!

Constraints on tuples
- **CHECK** condition
CONSTRAINTS ON ATTRIBUTES AND TUPLES

CREATE TABLE R (  
  A int NOT NULL,  
  B int CHECK (B > 50 and B < 100),  
  C varchar(20),  
  D int,  
  CHECK (C >= 'd' or D > 0))
CONSTRAINTS ON ATTRIBUTES AND TUPLES

CREATE TABLE Product (  
    productID CHAR(10),  
    name CHAR(30),  
    category VARCHAR(20),  
    price INT CHECK (price > 0),  
    PRIMARY KEY (productID),  
    UNIQUE (name, category)  
)
CREATE TABLE Purchase (  
  prodName CHAR(30)  
  CHECK (prodName IN  
         (SELECT Product.name  
          FROM Product)),  
  date DATETIME NOT NULL)  

What does this constraint do?

What is the difference from Foreign-Key?
CREATE ASSERTION myAssert CHECK
(NOT EXISTS(
  SELECT Product.name
  FROM Product, Purchase
  WHERE Product.name = Purchase.prodName
  GROUP BY Product.name
  HAVING count(*) > 200 )
)

But most DBMSs do not implement assertions
Because it is hard to support them efficiently
Instead, they provide triggers