software studio

OM to RDB: additional considerations

Daniel Jackson

For now, all rights reserved. Daniel Jackson, 2012.
problem 1: generalizations

suppose OM looked like this:

```
Date ! on Entry by User

Subject ! about Review content Text

Name

Category

Rating

Email

Password
```
identifying entities

how to handle Entry?

Date ! on Entry ! by User

Subject ! about Review ! content Text

Name ! name Category

Name ! name Category

Name ! name Category
generalizations: 3 options

1 table
› entries (id, type, by, on, name, rating, ...)

1 table/concrete entity
› subjects (id, by, on, name, ...)  
› reviews (id, by, on, rating, ...)

1 table/entity
› entries (id, by, on)
› subjects (id, name, ...)
› reviews (id, rating, ...)

considerations
› queries, nulls, need for type field
problem 2: abstract types

for primitive sets in OM
› type is often best viewed as abstract

examples
› email address
› review rating
› currency amount
› phone number

how to represent in RDB?
› have to just store rep
› not necessarily a big string
  payment as (amount, currency)
  phone as (areacode, phone, extension)
many to many

saw how to handle one-many relation
› put in table at many end

what about many-many?
› create a table for the relation

exercise: can you define the table for Approval?
opportunity: immutability

suppose you have an entity
› all relations but one immutable

then consider separating out
› into a fresh table

why?
› avoid read/write contention
› mutability suggests different role

users(username, since)
invitations(username, username)
responseRates(username, rate)