

35START

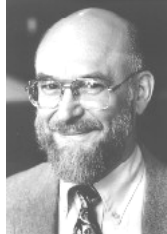
DAMA-Int'l, Denver, 2006/04




Subtypes & Supertypes

The Data Modeler's most Valuable Construct

©Gordon C. Everest
Professor Emeritus, Carlson School of Management
University of Minnesota



© Gordon C. Everest, All rights reserved.


431START

Advanced Database Design

2

7. Sub/Super Types

- “Abstractions” v. Collections v. Generalization
- Attribute, Relationship, & Entity Generalization
- Subtypes and Supertypes (Entity Generalization)
 - Underlying assumptions; conditions
 - Generalization vs. Specialization
 - Graphical representations
- Constraints
- Subtype Definition - distinguishing attribute
- Sub/SuperType Hierarchy/Lattice
 - The Universal Relation
- Inheritance (single; multiple) & Reuse
- Mapping to Tables

Gordon C. Everest
Carlson School of Management
University of Minnesota

© Gordon C. Everest, All rights reserved.

⊕

Starting from Basics

G

3 When you see this ER/schema diagram:

CUSTOMER

EMPLOYEE

What does it mean?

What can you assume?

What does "the system" assume?

N © Gordon C. Everest, All rights reserved.

⊕

"Abstractions" & Collections

4 SSTYPE Focusing on *selected* properties of objects

"Abstractions": (used in a different sense here)

- **CLASSIFICATION** ("Member-of")
 - Forming Types - entity sets/populations, domains
- **AGGREGATION** ("Part-of")
 - Building an entity record with descriptors (clustering attributes)
 - **COMPOSITION** (stronger "Part of" - no independent existence)
- **GENERALIZATION/SPECIALIZATION** ("Is-a")
 - Forming subtypes/supertypes, population subsets

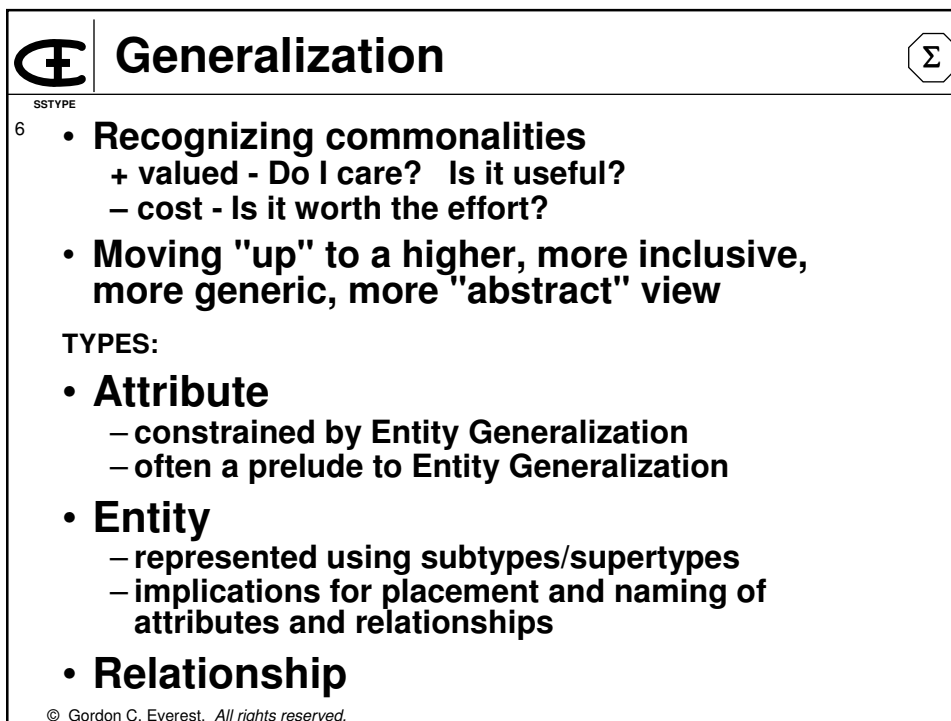
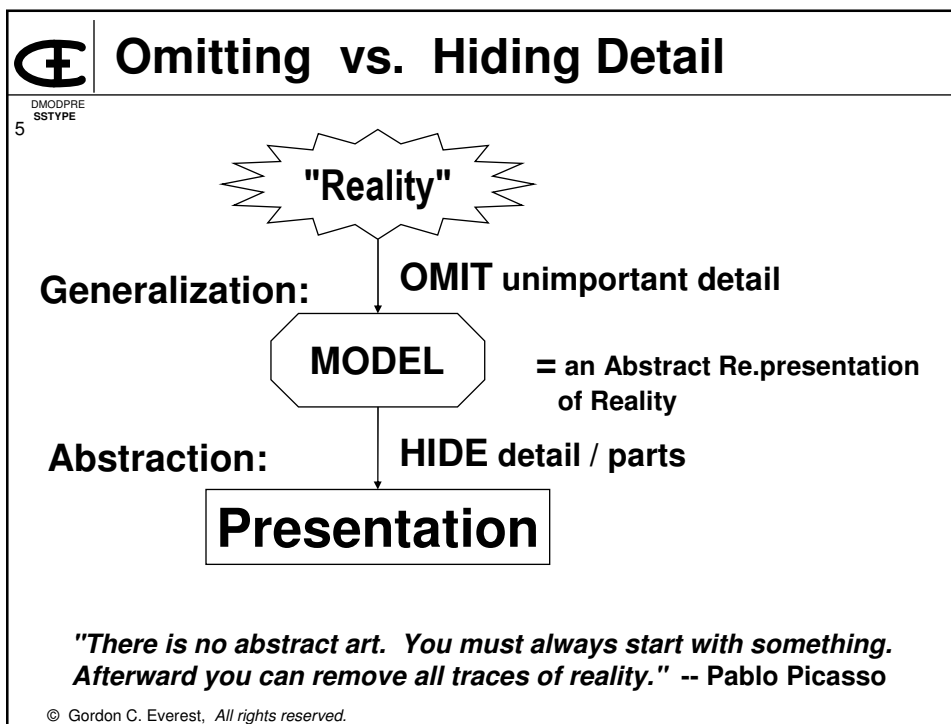
Collections: (assumes *homogeneous* members)


- **SET** – no duplicates and no order
- **BAG** – counting duplicates
- **SEQUENCE** – order matters
- ...

and looking at Relationships/Structure: GRAPH, TREE, ...

© Gordon C. Everest, All rights reserved.

-Batini, Ceri, & Navathe
- Smith & Smith (1977)
- Len Silverston
- Steve Hoberman
- David Hay





Attribute Generalization Examples

SSTYPE
Steve Hoberman, *Data Modeler's Workbench*

7

- **For a Tuxedo Rental shop, store Customer attributes:**
 - Waist size
 - Leg length
 - Neck size
 - Arm length
 - Shoulder width

=> Later add Shoe size.

What does that do to your database schema?

How might you solve the problem?

How does referential integrity become important here?


What is the down side of this schema redesign?

Similarly for Phone numbers:

Problems:

- Handling international numbers
- Handling other contact information, e.g. email

N © Gordon C. Everest, All rights reserved.



Attribute Generalization Example

SSTYPE
Steve Hoberman, *Data Modeler's Workbench*

8

Given the following three entity type populations:

What do you observe?

CUSTOMER	SUPPLIER	ASSOCIATE (Employee)
- First name	- Company name	- First name
- Last name	- Contact first name	- Last name
- Address line	- Contact last name	- Address line
- City	- Address line	- City
- State	- City	- State
- Zip code	- State	- Zip code
- Phone number	- Zip code	- Phone number
- Fax number	- Phone number	- Pager number
- Tax id	- Cell Phone Number	- Social Security #
- First order date	- Fax number	- Email address
- DUNS #	- Credit Rating	- Hire date
	- First PO Date	- Clock #
	- DUNS #	

**An ASSOCIATE can be assigned to several CUSTOMERs,
and manage the relationship with many SUPPLIERs.**

A CUSTOMER or SUPPLIER can contact multiple ASSOCIATES.

N © Gordon C. Everest, All rights reserved.

Attribute Generalization - Financial

SSTYPE

9 Suppose you saw a table defined like this:

How many rows would it have?

What would YOU want to do?

FINANCIAL DATA:

Dept	Year	Qtr	Bud/Act	Category	Amount
} Identifier					

A Classic Fact Table for a Dimensional Model!

How many rows would this table have?

FINANCIAL DATA:

- *Dept
- *Year
- Qtr1 Budget Material Amount
- Qtr2 Budget Material Amount
- Qtr3 Budget Material Amount
- Qtr4 Budget Material Amount
- Qtr1 Budget Labor Amount
- Qtr2 Budget Labor Amount
- Qtr3 Budget Labor Amount
- Qtr4 Budget Labor Amount
- Qtr1 Budget Capital Amount
- Qtr2 Budget Capital Amount
- Qtr3 Budget Capital Amount
- Qtr4 Budget Capital Amount
- Qtr1 Actual Material Amount
- Qtr2 Actual Material Amount
- Qtr3 Actual Material Amount
- Qtr4 Actual Material Amount
- Qtr1 Actual Labor Amount
- Qtr2 Actual Labor Amount
- Qtr3 Actual Labor Amount
- Qtr4 Actual Labor Amount
- Qtr1 Actual Capital Amount
- Qtr2 Actual Capital Amount
- Qtr3 Actual Capital Amount
- Qtr4 Actual Capital Amount

Extreme Attribute Generalization

SSTYPE

10

ENTITY

*EntityID

EntityName

←

ATTRIBUTE

*EntityID

*AttributeName

AttributeValue

How many rows?

What is lost here?

What is hard?

What is easy?

If find you are mixing value domains, you may have generalized too much.

ATTRIBUTE

*EntityID

*Name

Value

Type

Size

Precision

Units

LastUpdate

Source

Confidence

...

What else might be of interest about an attribute?

The Power of Generalization Thinking!

N © Gordon C. Everest, All rights reserved.

Understanding Subtypes/Supertypes: The Data Modeler's Most Important Construct

Gordon Everest, University of Minnesota

E

Relationship Generalization

SSTYPE
G. Simson, *DM Essentials*, 2005, p.140.

11 Reducing multiple relationships:

to one:

NOTE: We have already generalized the individual roles into a Person entity.

Where would you store the 'role' attribute?

How many foreign keys are stored? Where?

N © Gordon C. Everest, All rights reserved.

E

Model "Family Trees"

SSTYPE
Graeme Simson, *DM Essentials*, 2005


12 Fathers, Mothers, their Marriages, and Children:
What do these data models assume?



Generalized =>

Together with subtypes and supertypes:

... leaving open the question of where the Tables will be built.

N © Gordon C. Everest, All rights reserved.

	Generalization: Pros & Cons
	<p><small>SSTYPE</small></p> <p>13</p> <ul style="list-style-type: none">+ Fewer entities and relationships (simpler?)+ Greater flexibility to incorporate extensions+ Greater long-term stability of the model+ Looking for commonalities -> greater understanding+ handle special treatment of subsets- Hides business vocabulary<ul style="list-style-type: none">- business terms not in schema names but in attribute values- harder to express and enforce business rules or constraints. Most become conditional on the specialized subtype.- Problem defining the identifiers (Ref. modes)- queries are more difficult to formulate and less efficient to execute (more JOINS).<ul style="list-style-type: none">- e.g., "FIND Customers WHERE Waist = 42" - no longer works <p><small>© Gordon C. Everest, All rights reserved.</small></p>

	Entity Generalization					
	<p><small>SSTYPE</small></p> <p>14</p> <p>Discerning Inter-Entity Relationships:</p> <table border="0" data-bbox="487 1323 1071 1575"><tr><td>PERSON</td><td>ORGANIZATION</td></tr><tr><td>EMPLOYEE</td><td>SHAREHOLDER</td></tr><tr><td>CUSTOMER</td><td>VENDOR</td></tr></table> <p>Construct a high-level, conceptual data model.</p> <p><i>Problems?</i></p> <p><i>Observations?</i></p> <p><small>N</small></p> <p><small>© Gordon C. Everest, All rights reserved.</small></p>	PERSON	ORGANIZATION	EMPLOYEE	SHAREHOLDER	CUSTOMER
PERSON	ORGANIZATION					
EMPLOYEE	SHAREHOLDER					
CUSTOMER	VENDOR					

⊕

A Fundamental Assumption in a Data Model Diagram

SSTYPE

15

- The main construct is an Entity.
- Each labeled box/circle represents an Entity Type
 - a Defined Structure (a schema template)
 - a Population of Instances
- Grouping Instances into Types is essentially Arbitrary.
 - The world isn't naturally that way; the designer imposes a view
- All Entity Type Populations are strictly Disjoint (mutually exclusive; or non-overlapping).
 - At least that is the system's assumption, thus each file/table has its own set of records/tuples.

Is this always true?

What about: EMPLOYEE SHAREHOLDER

© Gordon C. Everest, All rights reserved.

⊕

Using Subtypes and Supertypes

SSTYPE Smith & Smith, ACM TODS, 1977/6.

16

- Subtypes and Supertypes allow us to formally represent overlapping populations
 - Every member of a subtype population is-a member of its supertype population(s).

so we can model:

- different roles played by members of a common population, e.g.:
 - role determines the attributes
- different states of an entity (over time), e.g.:

```

graph TD
    EMPLOYEE --> PERSON
    SHAREHOLDER --> PERSON
    CUSTOMER --> PERSON
    CUSTOMER --> ORGANIZATION
    
```

```

graph LR
    subgraph ORDER
    A[ORDER RECEIVED] -.-> B[ORDER VALIDATED & ACCEPTED]
    B -.-> C[ORDER FILLED]
    C -.-> D[BACK ORDER]
    D -.-> E[ORDER REJECTED]
    E -.-> A
    B -.-> E
    end
    
```

© Gordon C. Everest, All rights reserved.

☒ Subtype-Supertype "Relationship"

SSTYPE

17 • Tempting to call it a "Relationship"

```
graph LR; S([Supertype]) ---|1:1| T([Subtype]);
```

- $\text{pop}(\text{subtype}) \not\supseteq \text{pop}(\text{supertype})$
- **AND** the related members in the two sets are the *same* instance
 - that is what makes it different from relationships in a traditional ER/Relational data model, where the entity type populations are (assumed) disjoint!

© Gordon C. Everest, All rights reserved.

☒ Generalization / Specialization

SSTYPE

18 **Forming Entity Types**

- An “arbitrary” choice made by the Database Designer
- Recognizing when to use Subtypes and Supertypes
- Think about the entity *populations* you are modeling

Two basic and distinct situations:

- **Generalization:** (bottom-up - from several to a common supertype)
 - When you observe commonalities (e.g., common attributes*) across multiple entity populations.
 - the members may actually be from the same population, the same type of ‘thing’, so define a common supertype.
- **Specialization:** (top-down - from one to subtypes)
 - When there is something special about a subset of a population
 - They have some unique attributes*
 - You want to treat them differently
 - e.g., Apply a constraint, or have some attributes mandatory

*NOTE: speaking of attributes in ORM, means roles in relationships with other objects.

© Gordon C. Everest, All rights reserved.

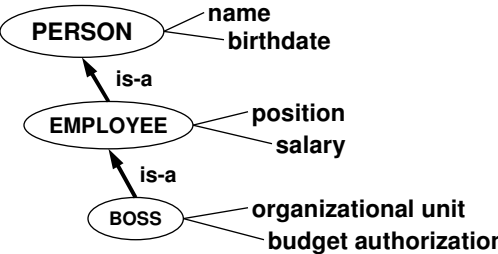
Σ

Subtypes and Supertypes

SSTYPE

19 **TWO CONDITIONS MUST ALWAYS BE TRUE:**

- each subtype population must be a subset (potentially) of each of its supertype populations
i.e., each instance of the subtype is in *every* supertype population
- each subtype inherits *all* the roles of its supertypes *and* must have additional roles/relationships



↑ More Instances
(larger population)

↓ More Attributes
/Roles
/Relationships

**If either condition is NOT true,
no reason to call out the subtype in a separate definition.**

© Gordon C. Everest, All rights reserved.

Σ

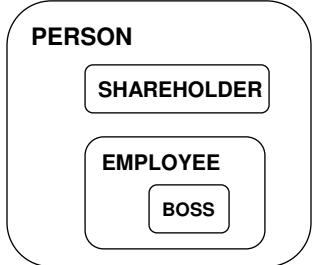
Diagramming Subtypes and Supertypes

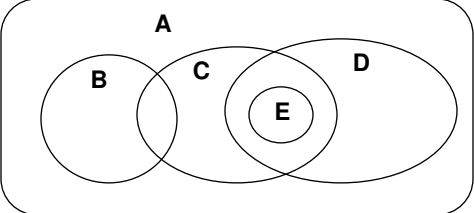
SSTYPE

20 Two Basic Representations:

1. NESTED (Euler Diagram)

- + Intuitive - visually shows inclusion
- + Clean and Compact
- Generally assumed disjoint
- Only good for simple cases
- Not good for complex cases - difficult to represent both exclusive and overlapping subtypes (like a Venn Diagram):





© Gordon C. Everest, All rights reserved.

Ⓔ

Diagramming Subtypes and Supertypes

SSTYPE

21 Two Basic Representations:

2. SEPARATED

- + More common
- + Easier to show constraints and multiple supertypes for more complex cases.
- not visually intuitive
- confusion with "relationship"
- Adds more "clutter"

ORM, UML

EER

IE

IDEF1X

© Gordon C. Everest, All rights reserved.

Ⓔ

Diagramming Exercise

G

SSTYPE

22 • Convert the following Nested diagram into a Separated S/Type diagram:

How to model 'E' ?

Any constraints required?

© Gordon C. Everest, All rights reserved.

⊆

Subtype / Supertype Constraints

SSTYPE

23

IN GENERAL, WITHOUT CONSTRAINTS, ASSUME:

- **overlapping** subtype populations
 - else **Disjoint**, so apply **Exclusion** constraint:
- **non-exhaustive** (Partial) on the supertype, i.e., a supertype instance need not be in *any* subtype
 - else **Mandatory/Totality/Dependency** constraint

=> Declare constraints on the more restrictive cases

- **Some systems allow only Disjoint and Total**
 - it is possible to model Overlapping, even if the system only allows Disjoint subtypes. *HOW?*
- **Some systems make Disjoint and Total the defaults**

N © Gordon C. Everest, All rights reserved.

⊆

Diagramming S/Type Constraints

SSTYPE

24

CONSTRAINTS (3 cases):

1. **Exclusive** or disjoint *subtypes* – X
2. **Exhaustive** or Total on the *supertype* – T
3. **Exhaustive** or Total on the *subtype* – T_b

e.g., MAN, WOMAN, CHILD

e.g., OVIPAROUS, MAMMAL, BIRD, FISH

© Gordon C. Everest, All rights reserved.

SSTYPE		S/Type Constraints - Other Notations			
25		<u>Exclusive or disjoint</u>	<u>overlapping</u>	<u>Exhaustive or Total</u>	<u>Partial</u>
NIAM	(X)			(T)	
ORM*	(X)	(default)		(•)	(default)
EER					
IE					
UML	(explicit English labels on the S/Type arcs)				
ODL					
IDEF1X					
*combined	(X)				
NOTE: No modeling schemes or systems recognize totality on the subtype.					
© Gordon C. Everest, All rights reserved.					

"Well-Defined" Subtypes

SSTYPE

26

- based on an attribute of the supertype
 - called the "distinguishing" attribute
- characteristics of the relationship determine the constraints on the subtypes
 - mandatory attribute => exhaustive/totally constraint
 - single-valued attribute => exclusive subtypes constraint

```

    graph TD
        Patient((Patient))
        Male((Male))
        Female((Female))
        Sex((Sex))
        Patient -- M:1 --> Sex
        Patient -.-> Male
        Patient -.-> Female
        Male -.-> Patient
        Female -.-> Patient
        Male -.-> Female
        Female -.-> Male
    
```

- What if an optional attribute?
- What if a multi-valued attribute (M:N relationship)?

© Gordon C. Everest, All rights reserved.

⊆

Subtype Definition

SSTYPE

27 **Attribute-Defined Subtype (Intentional Set)**

- a rule for including a Supertype instance in the Subtype
- Defined in terms of the values of a supertype attribute
- in general, a Boolean expression on attribute(s) of the supertype
- can be considered a *constraint* rule on subset membership
- there are many possible subgroupings (specializations) of an entity type based on the values of its attributes, so find those that matter.
- it is not always possible to define the rules for membership in a subtype, hence:

User-Defined Subtype (Extensional Set)

- inclusion determined by “existence” in the set; membership is manual, the system cannot automate or validate membership.
- Some systems *require* subtype definitions such as *VisioEA* (but... as free-form text!)
- Can always come up with an artificial distinguishing attribute

© Gordon C. Everest, All rights reserved.

⊆

Subscription Database (MIS Quarterly)

SSTYPE

28 **Current design**

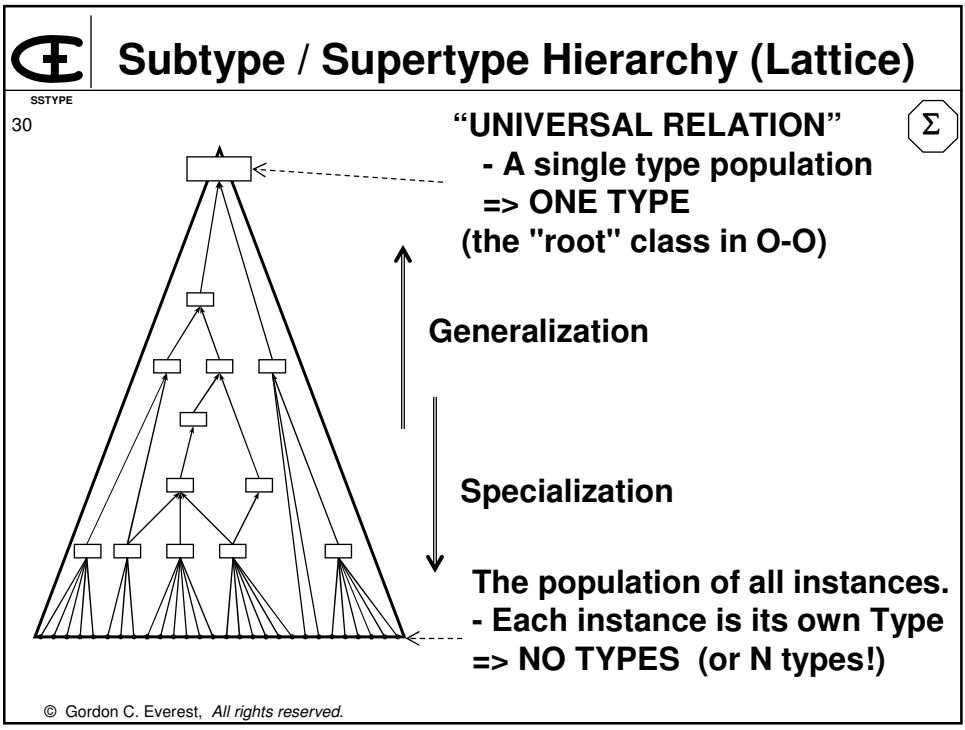
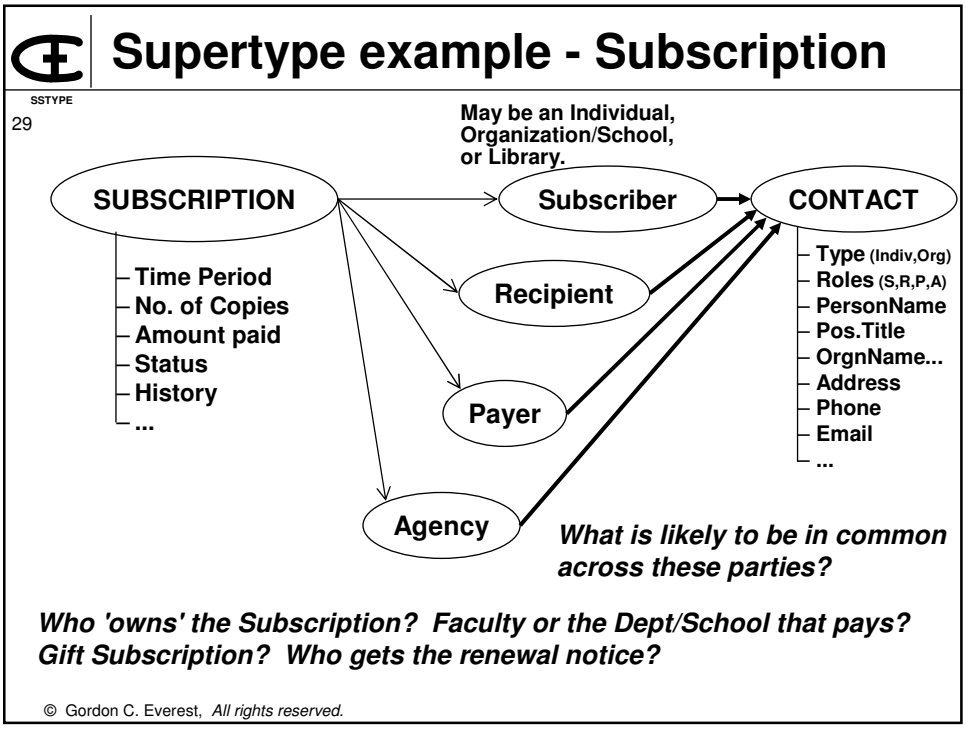
Active Entities: BASE Event

What commonalities do you observe?

© Gordon C. Everest, All rights reserved.

Understanding Subtypes/Supertypes: The Data Modeler's Most Important Construct

Gordon Everest, University of Minnesota





Sub/SuperType Hierarchy/Lattice - notes

SSTYPE

31 AT THE EXTREMES:

- a single supertype at the top is called the **UNIVERSAL RELATION**. If you built a single table for all the data in your organization:
 - *what would be the entity?*
 - *what would be the identifier?*
 - *what would be the attributes?*
 - *would all the attributes be relevant for each row?*

- at the bottom would be individual instances, each instance being its own type!
 - But sharing many attributes with other entities

The real art of database design is picking the appropriate entity types within the levels of the hierarchy.

- Allows the designer to defer choosing what tables to build

© Gordon C. Everest, All rights reserved.



Single vs. Multiple Inheritance

SSTYPE

32 • **SINGLE** - every subtype has only *one* supertype
=> a strict Hierarchy of types

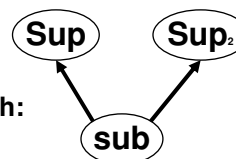
- **MULTIPLE** supertypes for a (Shared) Subtype

- If multiple supertypes, they must converge on one population higher up = the root type

=> a lattice.

so what is wrong/incomplete with:

- no lattice if no overlapping subtypes
- it is possible to transform multiple inheritance (lattice) to a generalization hierarchy by defining all possible combinations of subtypes
- each mini hierarchy or lattice has a root, all root objects are disjoint.



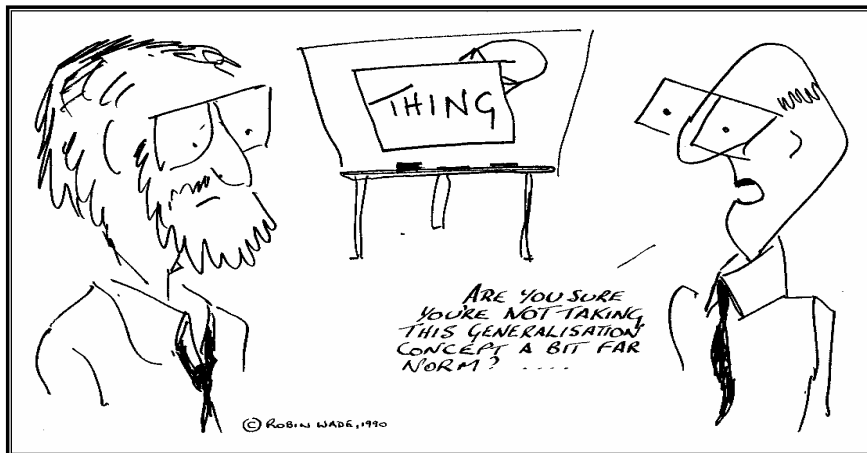
© Gordon C. Everest, All rights reserved.



Extreme Entity Generalization

SSTYPE

33



"The level of generalization is *critical*." - Graeme Simsion

What is the 'THING'?

SIMSION
BOWLES
& ASSOCIATES

© Gordon C. Everest, All rights reserved.



Inheritance and Reuse

SSTYPE

34 • Separate but related notions - often confused (Chris Date got it right)

DESIGN NOTION based on characteristics of populations:

- Multiple populations with some different characteristics sharing some common characteristics, ... so define a supertype (generalization).
- Need for special treatment of a subset of a population ... so define a subtype (specialization)
 - Subtype inherits common characteristics from its supertypes plus has some additional characteristics of interest


CONSTRUCTION - implementation efficiency => REUSE

- Inheritance of definitions of data and procedures
 - for efficiency of implementation
- SOLVING PROBLEMS:
 - overriding and blocking
 - static (copy at creation time only),
vs. dynamic (maintain linkage to automatically inherit changes)
 - multiple inheritance => conflict, priority order

© Gordon C. Everest, All rights reserved.


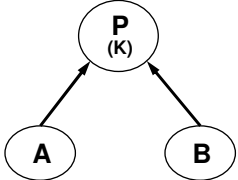
Understanding Subtypes/Supertypes: The Data Modeler's Most Important Construct

Gordon Everest, University of Minnesota

 Comparing Modeling Schemes							
SSTYPE		ER/Rel (Chen/Codd)	EER (Teorey**)	ORM (Halpin)	UML (OMG)	ODL (ODMG)	SQL 1999
35	Class Hierarchy	X	Y	Y	Y	Y	Y
	Disjoint*		Y	Y	default	only	only
	Overlapping		default	default	Y	X	X
	Total covering*		Y	Y	user-defined	on abstr.class	partly
	Partial		Y	Y	Y (incomplete)	Y	Y
	Attribute-defined discriminator		Y	'must' but...	limited (pseudo attrib)	X	X
	Shared Subclasses (multiple inheritance)		Y	Y	Y	X	X

*the more restrictive case, calling for a constraint declaration.

© Gordon C. Everest, All rights reserved.

 Mapping to (Relational) Tables		
36	<p>THREE BASIC CHOICES:</p> <ul style="list-style-type: none"> Supertype only: (Absorption - 'flatten' up) $\underline{K}_P D \{ P_i \} \dots \{ A_i \} \dots \{ B_i \} \dots$ Subtypes only: (not possible in VisioEA) (Separation - 'flatten' down) $\underline{K}_A \{ P_i \} \dots \{ A_i \} \dots$ $\underline{K}_B \{ P_i \} \dots \{ B_i \} \dots$ Both: (Partition) $\underline{K}_P D \{ P_i \} \dots$ $\underline{K}_A \{ A_i \} \dots$ $\underline{K}_B \{ B_i \} \dots$ 	<p>GIVEN: Halpin₀₁, p.426.</p>  <pre> graph TD A((A)) --> P((P(K))) B((B)) --> P </pre> <p>CONSIDER:</p> <ul style="list-style-type: none"> D = Distinguishing Attribute on P (optional) (single-valued?) Exclusive: on A & B - A & B overlapping Exhaustive (Total): P in neither A or B <p>PROBLEMS:</p> <ul style="list-style-type: none"> Redundancy Incomplete Querying

© Gordon C. Everest, All rights reserved.

Choosing a Mapping Strategy

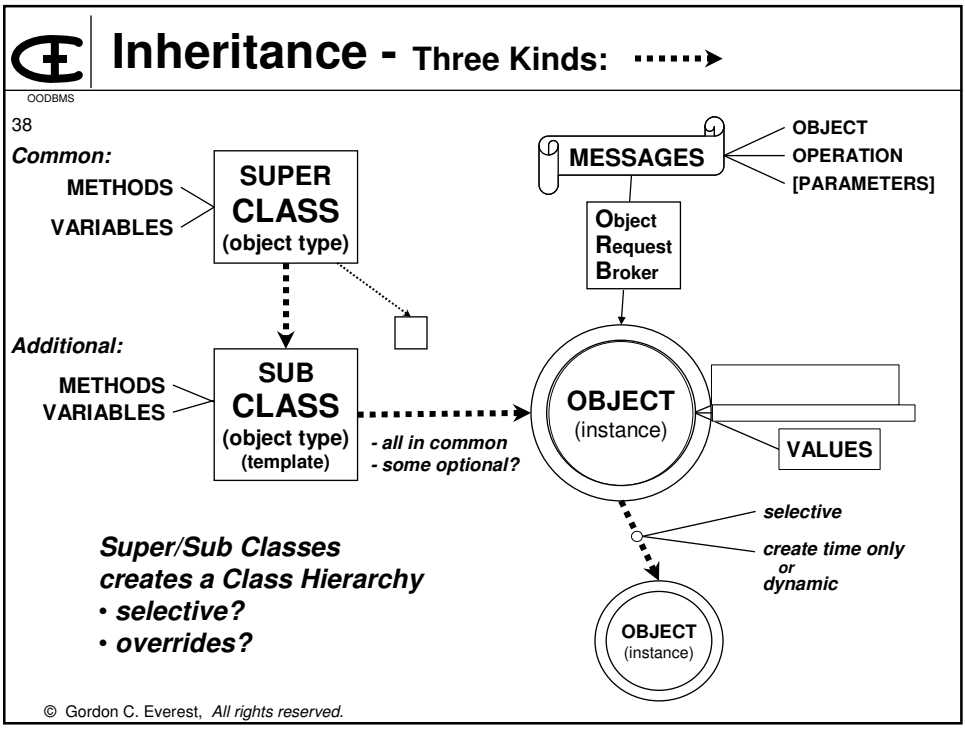
SSTYPE

37

MAPPING STRATEGY	Disjoint	Overlapping	Total	Partial	Queries
on Supertype (absorption) (flatten up)	nulls	multiple type D	+	more nulls	+
on Subtypes (separation) (flatten down)	+	<i>redundancy</i>	+	<i>arbitrary P</i> <small>(no place for P orphans)</small>	joins
on Both (partition)	+	+	+	+	more joins

on Sub is best if: #P >> #D

© Gordon C. Everest, All rights reserved.





Benefits of Subtype/Supertype

SSTYPE

G. Simson, *DM Essentials*, 2005, p.128ff.

- 39
- **Consciously and Creatively think...**
 - commonalities => generalization to supertype
 - special cases => specialization to subtypes
 - **Greater flexibility to handle extensions.**
 - **Greater stability for long-lived critical applications.**
 - **Generalization can reveal common patterns for reuse.**
 - **Abstraction for presentation, collapse the subtypes into their supertypes**
 - equivalent of "leveling" in process/DFD models
 - **Use subtyping to aid human understanding with no intention of implementing as separate tables.**
 - **Approach design top-down, bottom-up, or middle-out**
 - **Explicit representation of multiple table designs, thus deferring the choice for later implementation.**

© Gordon C. Everest, All rights reserved.



References

SSTYPE

- 40
- John SMITH and Diane P. SMITH, "Database Abstractions: Aggregation and Generalization," *ACM TODS*, (2:2) 1977. -classic
 - Chris J. DATE, "Subtypes and Supertypes: Setting the Scene," *Database Programming & Design*, 1999 February.
 - Terry HALPIN, *Information Modeling and Relational Databases*, Morgan Kaufmann, 2001, in ORM context.
 - Graeme C. SIMSION and Graham C. WITT, *Data Modeling Essentials*, 3e, Morgan Kaufmann, 2005, chap. 4.
 - Ramez ELMASRI and Shamkant NAVATHE, *Fundamentals of Database Systems*, 3e, Addison-Wesley, 2000, chapter 4 - academic, theoretical.
 - Steve HOBERMAN, *Data Modeler's Workbench*, Wiley, 2002, chapter 9 - practical with examples.
 - Susanne W. DIETRICH and Susan D. URBAN, *An Advanced Course in Database Systems - Beyond Relational Databases*, Prentice Hall, 2005.

© Gordon C. Everest, All rights reserved.