

# Databases, Information and Conceptual Schema Design

## Chapter 1

Prof. R. Meersman

*material adapted with permission from*

*Terry Halpin*

TerryHa@microsoft.com

www.orm.net

- ❖ Information systems modeling
- ❖ ORM (Object Role Modeling): conceptual schema design
- ❖ ER (Entity Relationship) and UML (Unified Modeling Language)
- ❖ ORM: mapping from conceptual to relational
- ❖ ORM: schema equivalence and optimization

# Information systems modeling

- The two fundamental skills
- Information systems development:
  - life cycle
  - perspectives, levels, worlds
  - frameworks
- Five language generations
- Information modeling methods

<i>Movie</i>	<i>Year</i>	<i>Director</i>	<i>Stars</i>
Awakenings	1991	Penny Marshall	Robert De Niro Robin Williams
Backdraft	1991	Ron Howard	William Baldwin Robert De Niro Kurt Russell
Cosmology	1994	Terry Harding	
Dances with wolves	1990	Kevin Kostner	Kevin Kostner Mary McDonnell

**An output report (external level)**

**Movie:**

<i>movieName</i>	<i>releaseYr</i>	<i>director</i>	<i>star</i>
Awakenings	1991	Penny Marshall	Robert De Niro
Awakenings	1991	Penny Marshall	Robin Williams
Backdraft	1991	Ron Howard	William Baldwin
Backdraft	1991	Ron Howard	Robert De Niro
Backdraft	1991	Ron Howard	Kurt Russell
Cosmology	1994	Terry Harding	?
Dances with wolves	1990	KevinKostner	Kevin Kostner
Dances with wolves	1990	KevinKostner	Mary McDonnell

*A relational database table.*

*Is anything wrong?*

**Movie:**

<i>movieName</i>	<i>releaseYr</i>	<i>director</i>
Awakenings	1991	Penny Marshall
Backdraft	1991	Ron Howard
Cosmology	1994	Terry Harding
Dances with wolves	1990	Kevin Kostner

**Starred:**

<i>movieName</i>	<i>star</i>
Awakenings	Robert De Niro
Awakenings	Robin Williams
Backdraft	William Baldwin
Backdraft	Robert De Niro
Backdraft	Kurt Russell
Dances with wolves	Kevin Kostner
Dances with wolves	Mary McDonnell



The implemented database design should:

- *correctly* model your business
- *completely* model those aspects of interest
- *efficiently* model your business

*Uncontrolled* redundancy can lead to  
inconsistency  
inefficiency

*Movie* ( movieName, releaseYr, director )

*Starred* ( movieName, star )



Consider the following English query:

For each movie, list its name, director  
and stars (if any).

How can we say this in *SQL*  
(a relational query language)?

**Movie:**

<i>moviename</i>	<i>releaseyr</i>	<i>director</i>
Awakenings	1991	Penny Marshall
Backdraft	1991	Ron Howard
<b><i>Cosmology</i></b>	<b><i>1994</i></b>	<b><i>Terry Harding</i></b>
Dances with wolves	1990	Kevin Kostner

**Starred:**

<i>movieName</i>	<i>star</i>
Awakenings	Robert De Niro
Awakenings	Robin Williams
Backdraft	William Baldwin
Backdraft	Robert De Niro
Backdraft	Kurt Russell
Dances with wolves	Kevin Kostner
Dances with wolves	Mary McDonnell

```
select Movie.movieName, director, star
from Movie, Starred
where Movie.movieName = Starred.movieName
```

*Is this OK?*

**Movie:**

<i>movieName</i>	<i>releaseYr</i>	<i>director</i>
Awakenings	1991	Penny Marshall
Backdraft	1991	Ron Howard
<b><i>Cosmology</i></b>	<b><i>1994</i></b>	<b><i>Terry Harding</i></b>
Dances with wolves	1990	Kevin Kostner



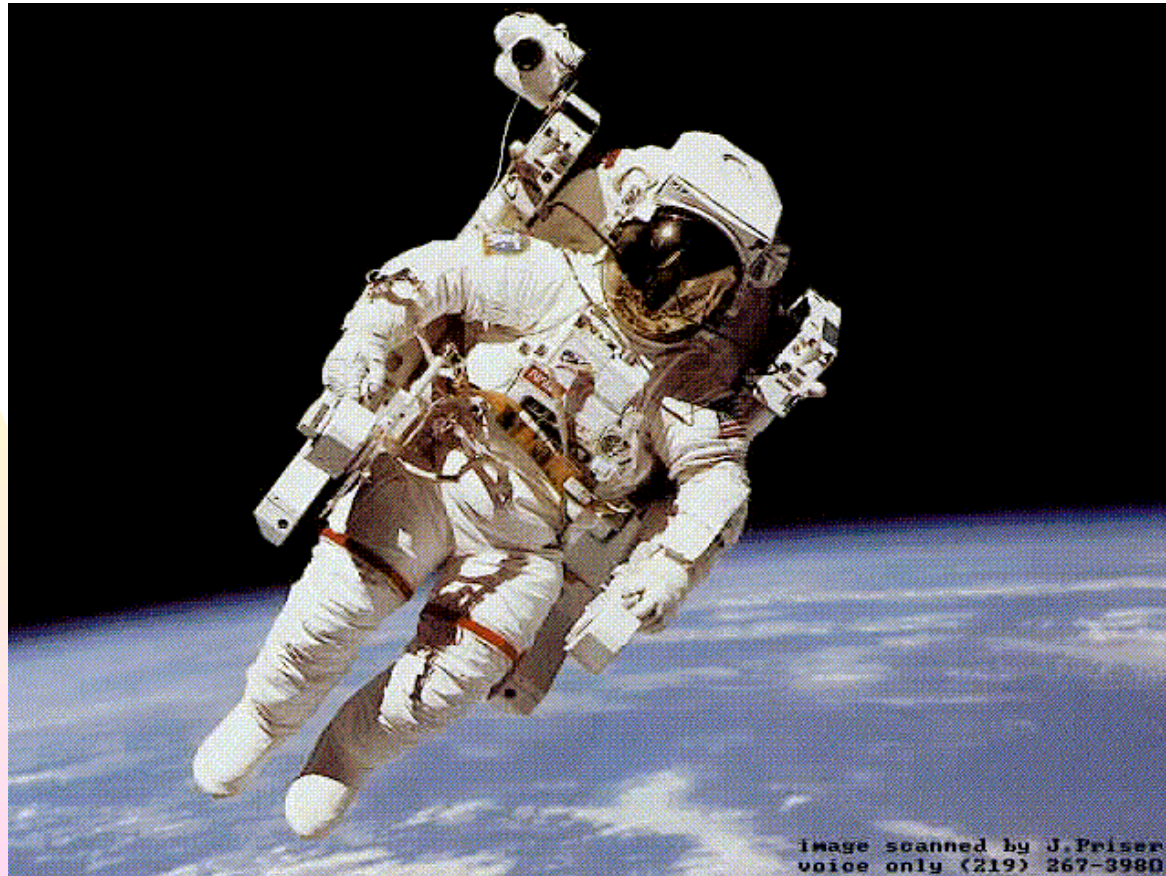
**Starred:**

<i>movieName</i>	<i>star</i>
Awakenings	Robert De Niro
Awakenings	Robin Williams
Backdraft	William Baldwin
Backdraft	Robert De Niro
Backdraft	Kurt Russell
Dances with wolves	Kevin Kostner
Dances with wolves	Mary McDonnell

```
select Movie.movieName, director, star
from Movie left outer join Starred
on Movie.movieName = Starred.movieName
```

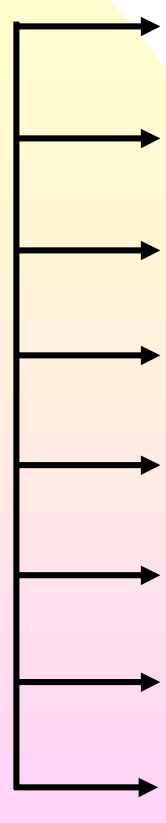
# Two fundamental human skills

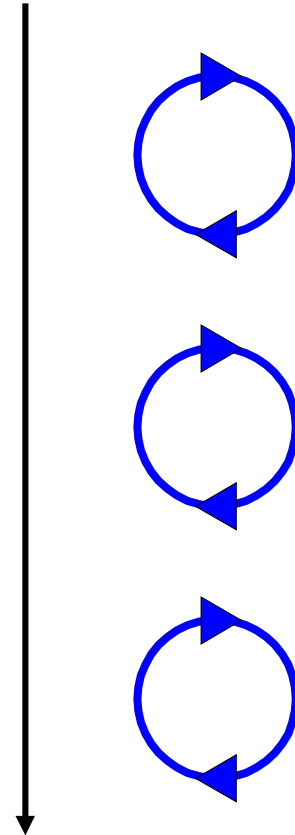
- *describe* the universe of discourse
- *query* the system about the UoD



“ORM lifts our thinking up so we can model and query the world at the conceptual level” [Halpin 2000]

## Procedure: *Information systems life cycle*

- 
- Feasibility study
  - Requirements analysis
  - **Conceptual design (data, process)**
  - **Logical design (data, process)**
  - External design (data, process)
  - Prototyping
  - Internal design and implementation
  - Testing, validation and maintenance



*Large projects are often developed iteratively*

This high level cycle can be fleshed out with many detailed procedures for each phase (e.g. ORM CSDP).

The rise of component technology, design patterns and ERP packages has seen a move from

“Build or Buy”

to

“Build and Buy and Reuse”<sup>1</sup>

---

<sup>1</sup>e.g. see ERP articles in *CACM*, April 2000

## 2 main Perspectives: *Data* *Process*<sup>1</sup>

-- *what* information

-- *how* it is used

In the 70s, IS modeling was typically process-driven.

But for database applications, data structures were more central and more stable than processes. And refining processes into data-structures was too laborious.

So data-driven approaches became more widely used.

Current modeling approaches are often partly process-driven and partly data-driven. Data and process modeling can be done in tandem or parallel, with cross-checks for consistency and completeness.

---

<sup>1</sup> The following IFIP study added *Behavior* as 3<sup>rd</sup> perspective, but with no clear distinction between process and behavior: Olle et al. 1991, *Information Systems Methodologies: a Framework for Understanding*, 2<sup>nd</sup> edn, Addison-Wesley.

## 4 Levels: 4 schemas

- 1. Conceptual* human-oriented  
natural language  
elementary facts  
implementation-independent  
most stable  
e.g. true ER, ORM, some of UML
- 2. Logical* commit to underlying model type  
e.g. relational, hierarchic, network,  
ODMG Object model,  
SQL:1999 Object-Relational  
compound facts (normally)  
DBMS-independent (within model type)
- Many commercial ERs are mainly at this level,  
e.g. IDEF1X

### *3.Internal*

physical implementation model actually used to  
store data

execute operations

(add, delete, edit, read, navigate)

indexes, pointers, clustering etc.

e.g. an SQL Server database

### *4.External*

user interfaces

e.g. forms, reports for a specific user-group

interfaces may differ in:

content (data, operations)

display (format, help etc.)

usage (mouse, key etc.)

## 4 worlds<sup>1</sup>

*Subject world*

universe of discourse (UoD)  
= application domain

*System world*  
 (“dry”)

the information system’s formal model  
of the application domain

*Usage world*  
 (“wet”)

the world in which the system is to function,  
including user community, user interfaces etc.  
(a.k.a. environment of discourse)

*Development world*

environment and processes used to develop  
the system, including modelers, programmers,  
design methods, project schedules etc.

---

<sup>1</sup>Jarke M., Mylopoulos J. & Vassiliou, Y. 1992, ‘DAIDA: an environment for evolving information systems’, *ACM TODS*, vol. 10, no. 1, pp. 1-50.

## “6 perspectives”

“I keep six honest serving friends ...  
Their names are

*What*  
and  
*Why*  
and  
*When*  
and  
*How*  
and  
*Where*  
and  
*Who”*

---

<sup>1</sup>Kipling, R. 1902, “The Elephant’s Child”

# Levels + perspectives

A number of 2-dimensional frameworks have been proposed using the dimensions of Level and Perspective.

The most influential has been the Zachman Framework<sup>1</sup>. This has 5 levels and 6 perspectives, resulting in 30 cells. It is promoted by the Zachman Institute for Framework Advancement (ZIFA)<sup>2</sup>.

Any systems engineering project needs to consider:  
Which cells are relevant?  
How do we map between levels?

---

<sup>1</sup>Zachman, J. 1987, 'A framework for information systems architecture', *IBM Systems Journal*, vol. 26, no. 3.

<sup>2</sup>[www.zifa.com](http://www.zifa.com)

# Zachman Framework for Enterprise Architecture

	data <i>what</i>	function <i>how</i>	network <i>where</i>	people <i>who</i>	time <i>when</i>	motivation <i>why</i>
scope						
enterprise model						
system model						
technology model						
detailed representation						

## *Level/Focus: 4x6*

	<i>Data</i> <i>(what)</i>	<i>Process</i> <i>(how)</i>	<i>Deployment</i> <i>(where)</i>	<i>People</i> <i>(who)</i>	<i>Time</i> <i>(when)</i>	<i>Motivation</i> <i>(why)</i>
<i>Conceptual</i>						
<i>Logical</i>						
<i>Physical</i>						
<i>External</i>						

Such a 2-D grid is useful as an organizational overview.

Add guidelines for deciding:

- which cells to use?
- which tools to use for each cell?
- which tools to use for mapping?

# Abstraction

For a given purpose:

Show all relevant detail

Hide unwanted detail

Levels and perspectives are abstraction mechanisms.

There are many others,

e.g.


step-wise refinement (e.g. within a cell)

object-type zoom

paging

layering/toggles

componentization



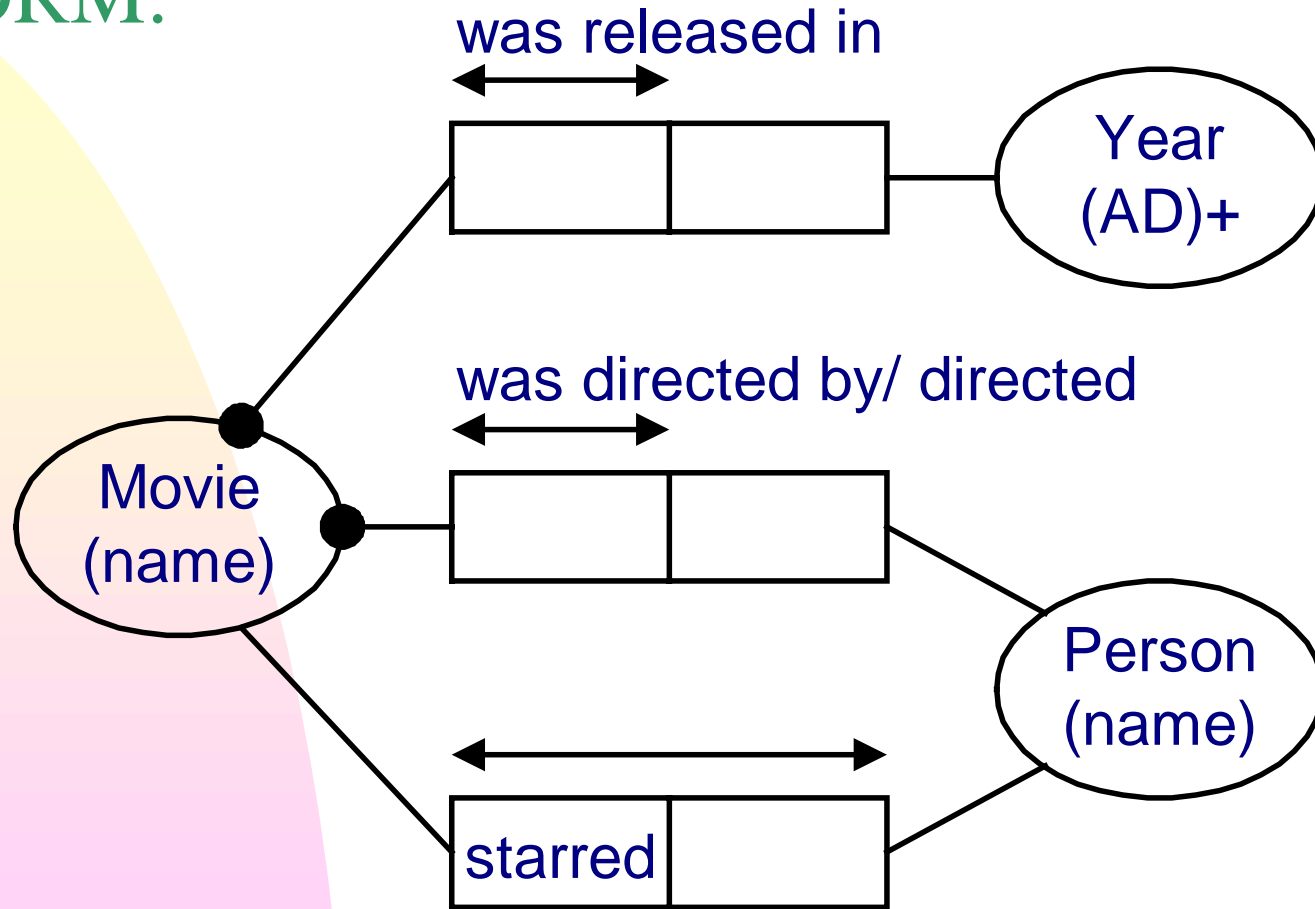
The data model and query design should be:  
performed at the conceptual level;  
then mapped to the other levels.

This vastly simplifies:  
getting it right in the first place;  
making changes later as the business evolves

<i>Movie</i>	<i>Year</i>	<i>Director</i>	<i>Stars</i>
Awakenings	1991	Penny Marshall	Robert De Niro Robin Williams
Backdraft	1991	Ron Howard	William Baldwin Robert De Niro Kurt Russell
Cosmology	1994	Terry Harding	
Dances with wolves	1990	Kevin Kostner	Kevin Kostner Mary McDonnell

**external level**

# ORM:



## RDB:

*Movie* ( movieName, releaseYr, director )



*Starred* ( movieName, star )

## ORDB:

*Movie* ( movieName, releaseYr, director, {star} )

English:

*For each movie, list its name,  
who directs it  
and who (if anybody) stars in it.*

ConQuer:

✓ Movie

└─ was directed by ✓ Person<sub>1</sub>

└─ **possibly** starred ✓ Person<sub>2</sub>

RIDL:

for Movie list (Name of it, Person  
directing it, Person starring\_in it)

# 5 generations of computer languages

<i>Generation</i>	<i>Language example</i>	<i>Sample code for same task</i>
5	ConQuer	✓Planet <b>that</b> has ✓Mass <b>and possibly</b> is orbited by ✓Moon
4	SQL	<b>select</b> X1.planetName, X1.mass, X2.moonName <b>from</b> Planet <b>as</b> X1 <b>left outer join</b> Moon <b>as</b> X2 <b>on</b> X1.planetName = X2.planetName
3	Pascal	Two pages of instructions like: <b>for</b> i := 1 <b>to</b> n <b>do begin</b> write (planetName[i], mass[i]);
2	8086 Assembler	Many pages of instructions like: ADDI AX, 1
1	8086 machine code	Many pages of instructions like: 00000101 00000001 00000000

# Information modeling methods

**Method** = Language(s) + Procedure(s)

<i>Language (Notation)</i>		
<i>Informal</i>	<i>Structured</i>	<i>Formal</i>
No formal syntax  e.g. cartoons natural language	Formal syntax  e.g. most DFDs some DSDs	Formal syntax and Formal semantics  e.g. predicate calculus Petri nets Z FORML, ConQuer

<i>Language</i>	
<i>Graphical</i>	<i>Textual</i>
Intuitive	Sometimes natural
Fast communication	Slower communication
Limited expressive power	Usually more expressive

Often best to use a combination of languages

*Design procedures:*

Very important in practice

Too often neglected

# Conceptual modeling language criteria

- ◆ Expressibility
- ◆ Clarity
- ◆ Simplicity and orthogonality
- ◆ Semantic stability
- ◆ Semantic relevance
- ◆ Validation mechanisms
- ◆ Abstraction mechanisms
- ◆ Formal foundation

## ER and UML

## ORM

World  
view

entities that  
have attributes and  
participate in  
relationships

objects that  
play roles

Graphical  
language

rectangles  
diamonds and lines  
etc.

ellipses  
role boxes  
etc.

ORM 's avoidance of attributes has many advantages, e.g. stability, populability, verbalization, no null values

Modeling *perspectives*:            *data*                            (what facts)  
   *process*                        (how)  
   ...

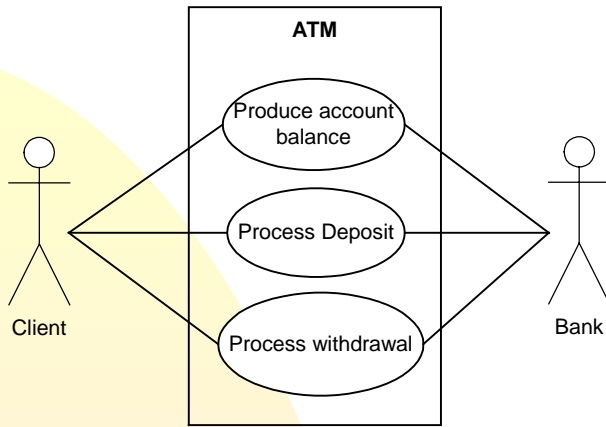
OO approaches often claim to unify data and process modeling.  
They have not yet done so  
e.g. weak links between use cases, classes and activities.

Modeling *method* = *notation (language)* +  
*procedures* for developing models.

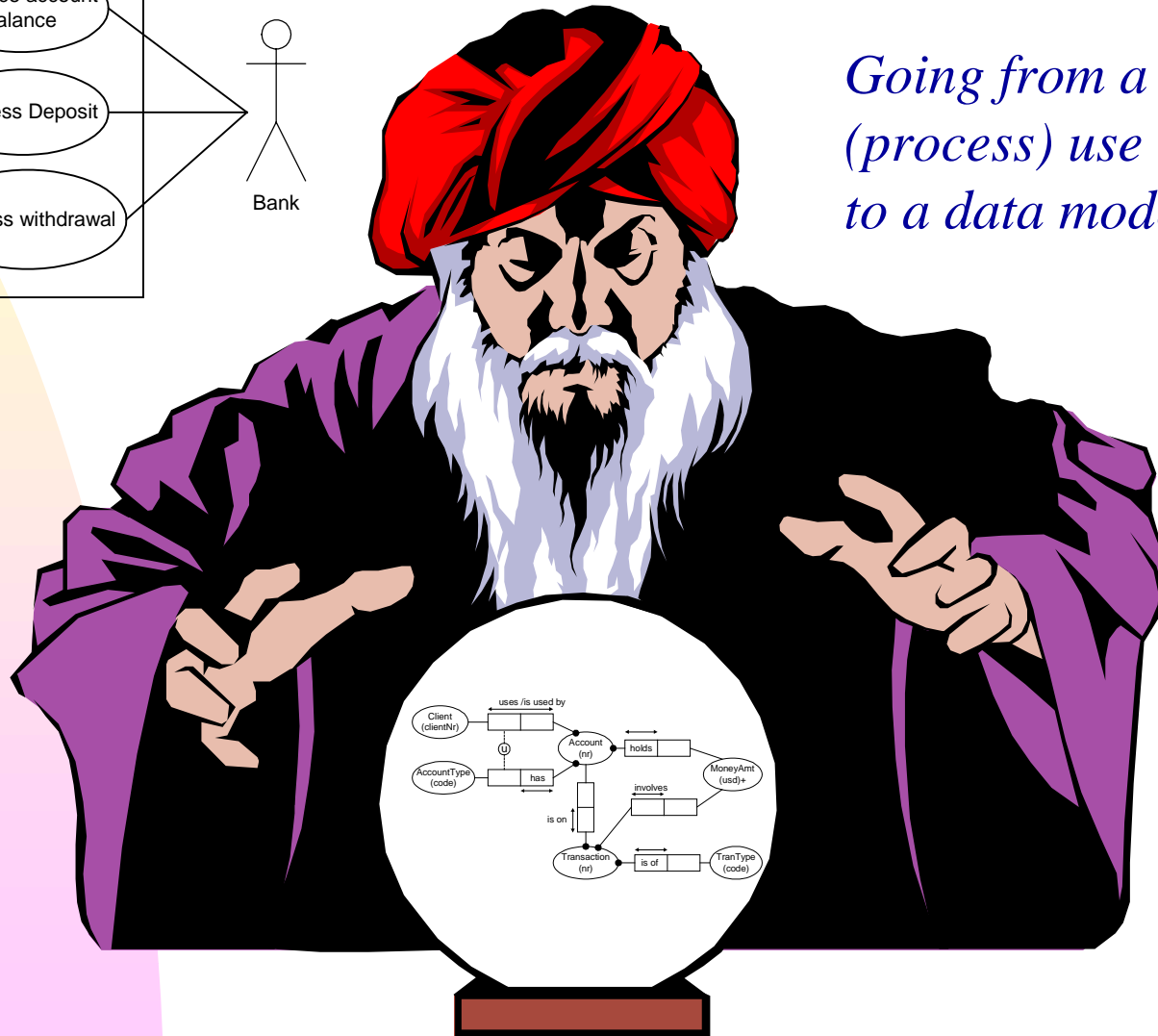
UML advocates a                    “use-case driven,  
   architecture-centric,  
   iterative approach”.

# UML use cases are ...

- ◆ Cases of the information system being used
- ◆ Useful for requirements analysis
- ◆ Typically focused on process modeling
- ◆ Offer only limited help for data modeling



*Going from a  
(process) use case  
to a data model*



*For data modeling we need “data use cases”*

*i.e. cases (examples) of data being used*

*e.g. sample output reports, input forms/screens*

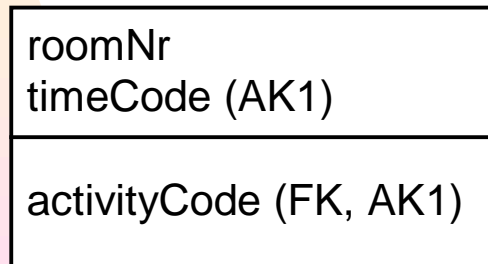
<i>Room</i>	<i>Time</i>	<i>ActivityCode</i>	<i>ActivityName</i>
20	Mon 9 am	VMC	VisioModeler class
20	Tue 2 pm	VMC	VisioModeler class
33	Mon 9 am	AQD	ActiveQuery demo
33	Fri 5 pm	SP	Staff party
...	...	...	...

*How do we model this report?*

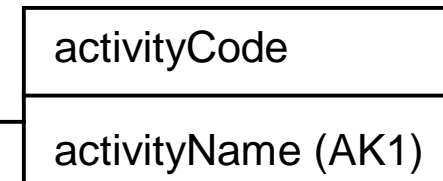
<i>Room</i>	<i>Time</i>	<i>ActivityCode</i>	<i>ActivityName</i>
20	Mon 9 am	VMC	VisioModeler class
20	Tue 2 pm	VMC	VisioModeler class
33	Mon 9 am	AQD	ActiveQuery demo
33	Fri 5 pm	SP	Staff party
...	...	...	...

## IDEF1X

RoomSchedule

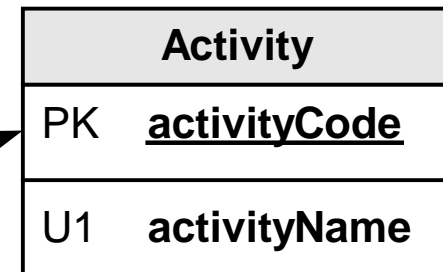
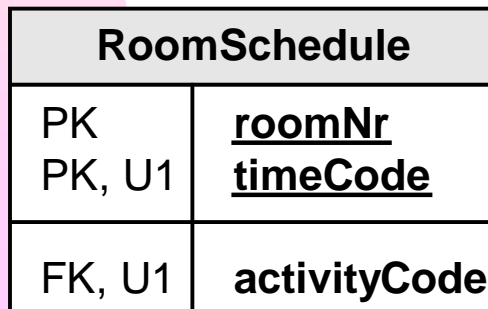


Activity



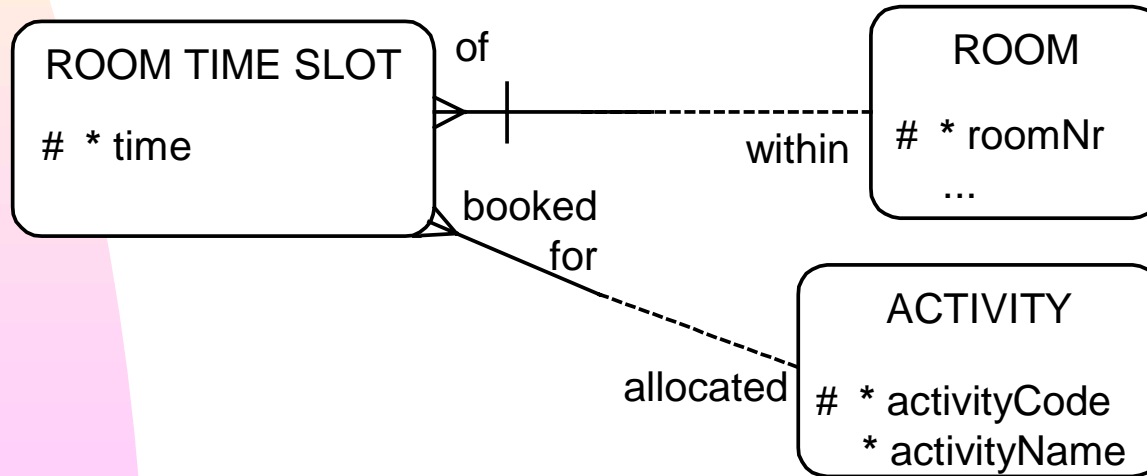
has /  
is of

## Relational



<i>Room</i>	<i>Time</i>	<i>ActivityCode</i>	<i>ActivityName</i>
20	Mon 9 am	VMC	VisioModeler class
20	Tue 2 pm	VMC	VisioModeler class
33	Mon 9 am	AQD	ActiveQuery demo
33	Fri 5 pm	SP	Staff party
...	...	...	...

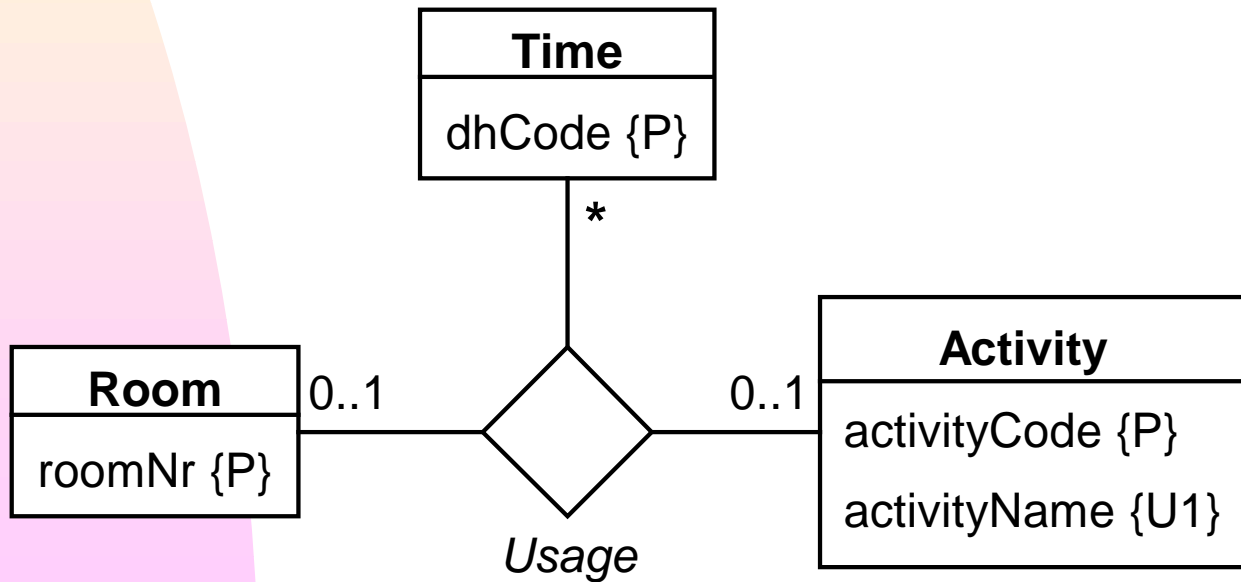
## Barker ER



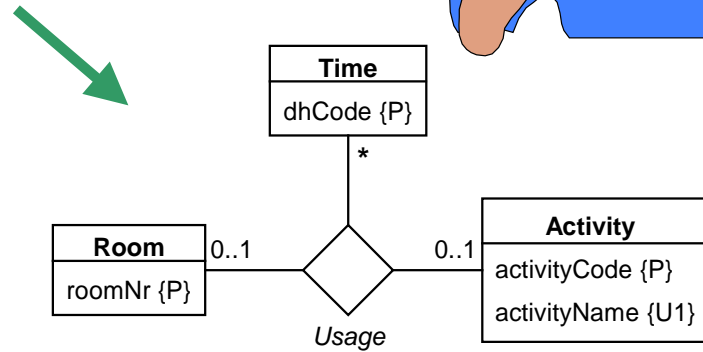
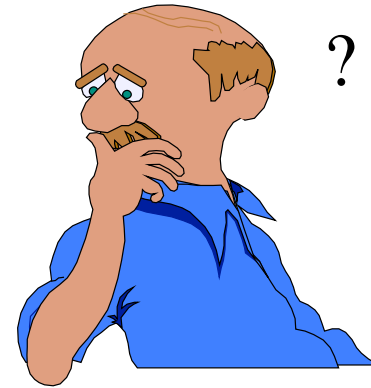
*Missing constraints?*

<i>Room</i>	<i>Time</i>	<i>ActivityCode</i>	<i>ActivityName</i>
20	Mon 9 am	VMC	VisioModeler class
20	Tue 2 pm	VMC	VisioModeler class
33	Mon 9 am	AQD	ActiveQuery demo
33	Fri 5 pm	SP	Staff party
...	...	...	...

## UML



<i>Room</i>	<i>Time</i>	<i>ActivityCode</i>	<i>ActivityName</i>
20	Mon 9 am	VMC	VisioModeler class
20	Tue 2 pm	VMC	VisioModeler class
33	Mon 9 am	AQD	ActiveQuery demo
33	Fri 5 pm	SP	Staff party
...	...	...	...



We need a practical procedure for

- abstracting from data use cases to the data model
- validating the data model



The *domain expert* best understands the application domain, so should play an active role in modeling sessions.

The *modeler*'s main task is to formalize the domain expert's informal knowledge.

Communication  
between modeler  
and domain expert  
should:



- Verbalize fact instances from data use cases
- Verbalize fact types in natural language
- Validate rules in natural language
- Validate rules using sample populations

# Object Role Modeling (ORM)

- ◆ Understandable Express facts and rules in English and/or intuitive graphics
- ◆ Reliable Validate rules using English and sample populations (multiple instances)
- ◆ Expressive Capture more business rules graphically
- ◆ Stable Minimize the impact of change to models and queries

*ORM may be used instead of, or prior to, other approaches*

# Fact Orientation

- ◆ Data examples are expressed as elementary facts in English sentences.
- ◆ How facts are grouped into structures (e.g. R-tables, OR-tables, Object classes) is not a conceptual issue.
- ◆ Fact-orientation is at a conceptual level above object-orientation.

*This is where modeler and domain expert should communicate and validate their understanding.*

# Fact structure notations

<i>ORM</i>	<i>UML</i>	<i>ER</i>
Object type: Entity type	Object class	Entity type
Value type	Data type	Value type
-- *	<i>Attribute</i>	<i>Attribute</i>
Unary relationship	-- **	--**
2 <sup>+</sup> -ary relationship	Association	Often binary only
Objectified relationship	Association class	--
Co-reference	Qualified assoc. §	--

\* use relationship

\*\* use Boolean attribute

§ incomplete coverage

# Role and Object: unifying concepts

- ◆ ORM's concept of a *Role* unifies UML's concept of *AssociationEnd* with the notion of an *attribute*, resulting in a notation that is:
  - simpler
  - more orthogonal
  - more expressive
- ◆ ORM's concept of *Object* unifies entities and values

# “The Telephone Heuristic”

<i>Room</i>	<i>Time</i>	<i>ActivityCode</i>	<i>ActivityName</i>
20	Mon 9 am	VMC	VisioModeler class
20	Tue 2 pm	VMC	VisioModeler class
33	Mon 9 am	AQD	ActiveQuery demo
33	Fri 5 pm	SP	Staff party
		...	...



## ORM CSDP

Step 1a: verbalize (domain expert)

Step 1b: verbalize as elementary facts  
(modeler)

# Ternary Facts

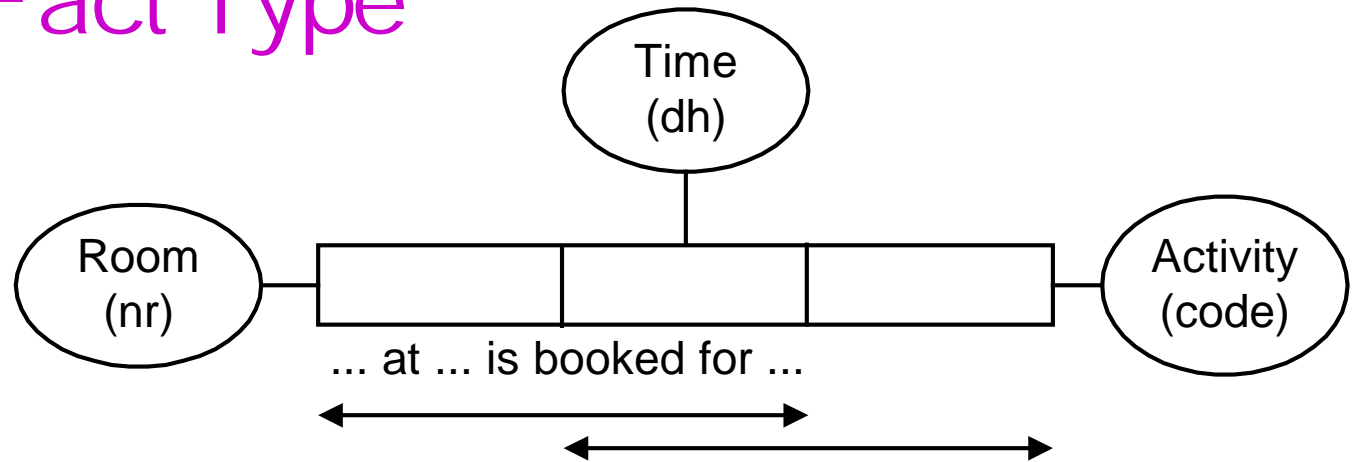
<i>Room</i>	<i>Time</i>	<i>ActivityCode</i>	<i>ActivityName</i>
20	Mon 9 am	VMC	VisioModeler class
...	...	...	...

*Fact 1:* Room '20'  
at  
Time 'Mon 9am'  
is booked for  
Activity 'VMC'

*Fact type:* Room at Time is booked for Activity

*Predicate:* ...at ... is booked for ...

# Ternary Fact Type



*Sample population  
(positive):*

20	Mon 9am	VMC
20	Tue 2 pm	VMC
33	Mon 9 am	AQD
33	Fri 5pm	SP
...	...	...

*Verbalize constraints (+ve):*

**Given any Room and Time,**

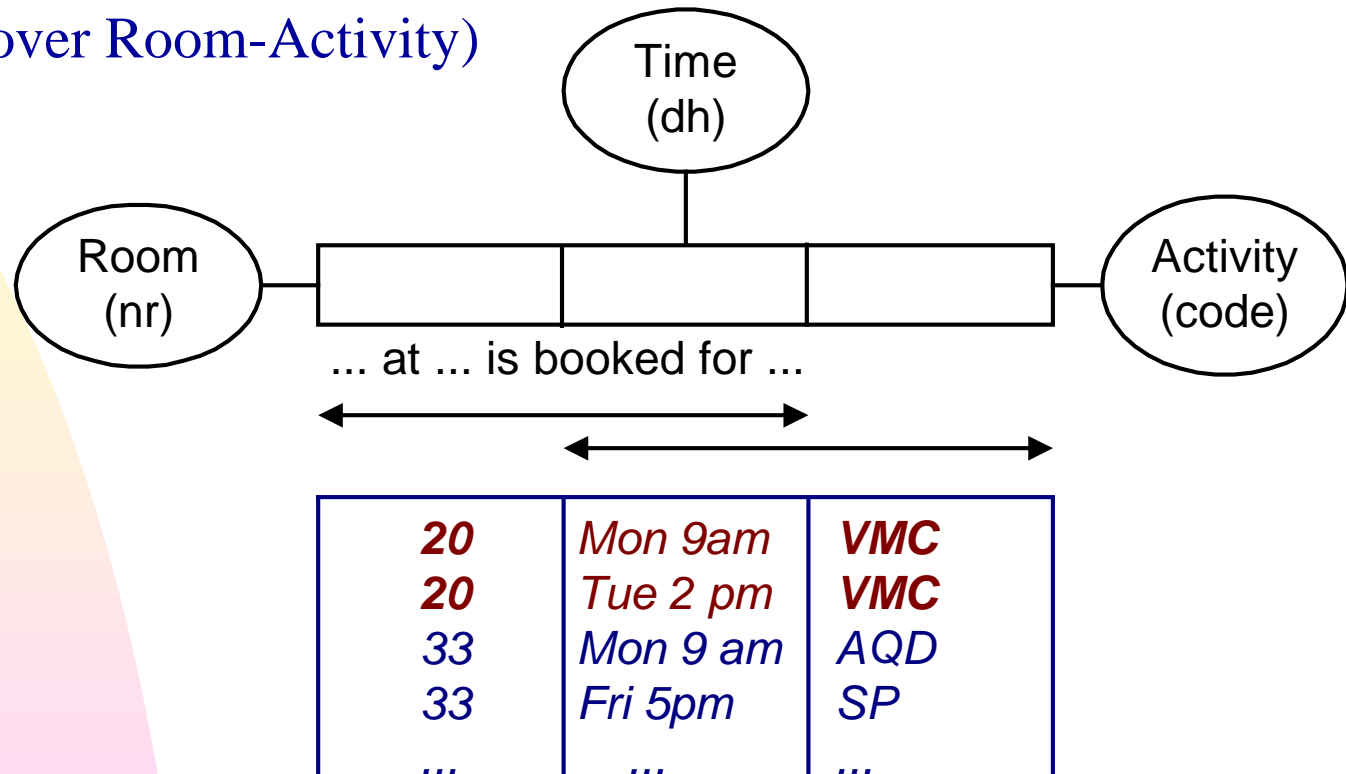
**that Room at that Time is booked for at most one Activity.**

**Given any Time and Activity,**

**at most one Room at that Time is booked for that Activity.**

*Verbalize default (lack of constraint):*

*(no UC over Room-Activity)*

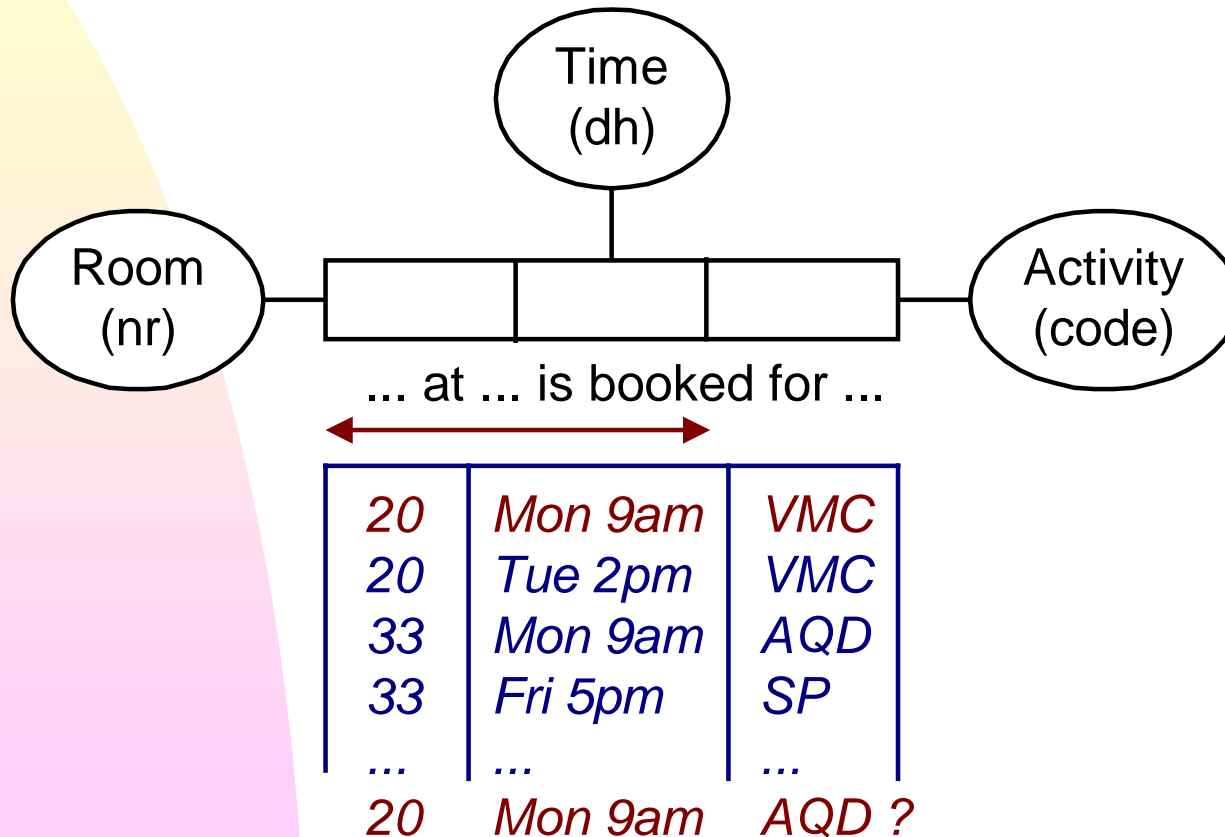


**it is possible that**

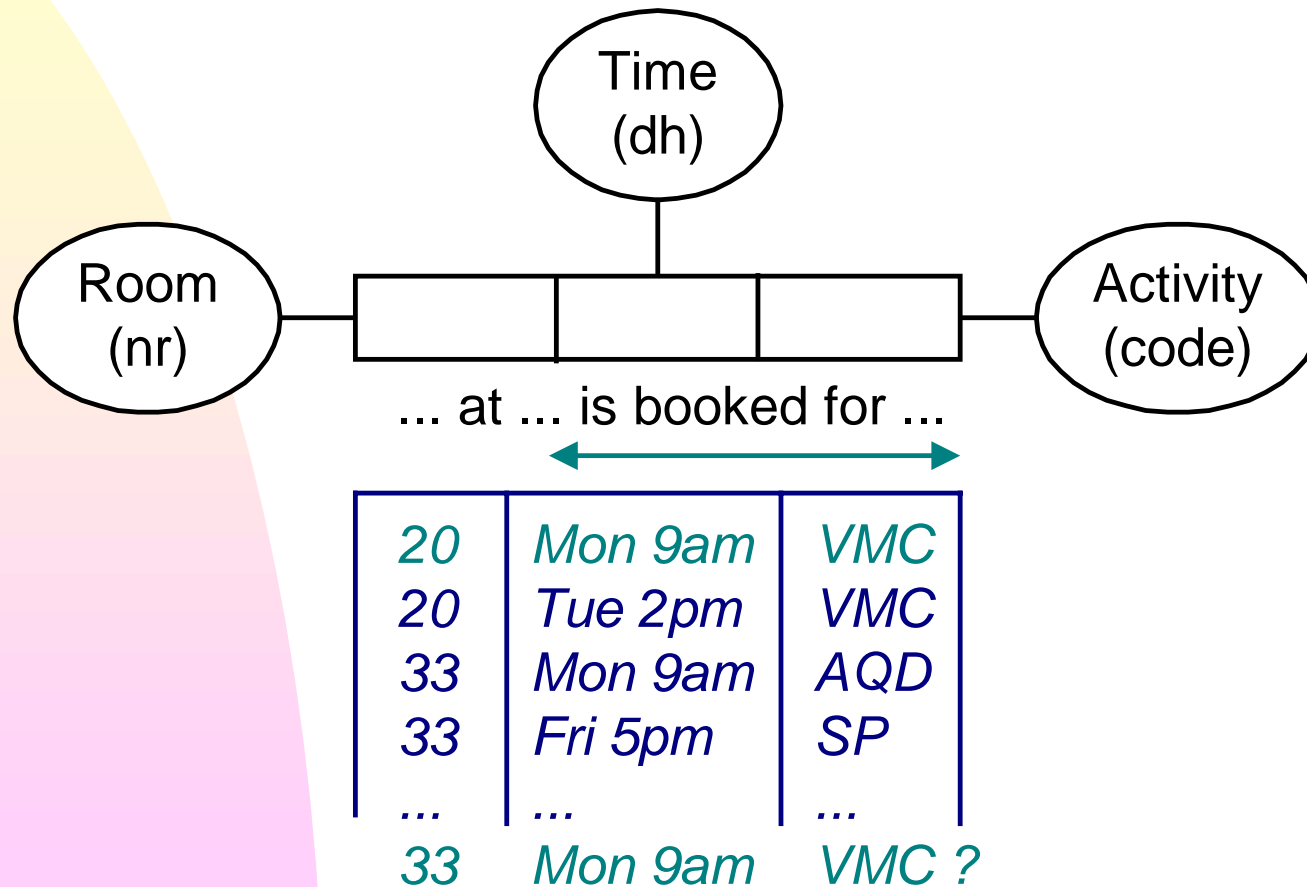
**the same Room at more than one Time is booked for the same Activity**

*Negative verbalization and counter-examples:*

**It is impossible that the same Room at the same time Time is booked for more than one Activity**



It is impossible that more than one Room at the same Time is booked for the same Activity



# Binary Facts

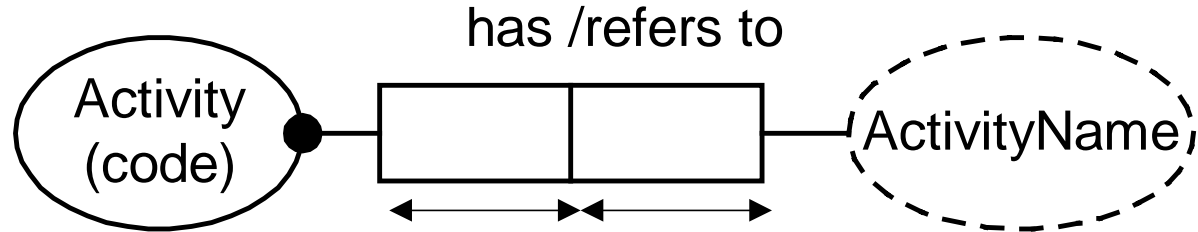
<i>Room</i>	<i>Time</i>	<i>ActivityCode</i>	<i>ActivityName</i>
20	Mon 9 am	VMC	VisioModeler class
...	...	...	...

*Fact 2:*            Activity ‘VMC’  
                         has  
                         ActivityName ‘VisioModeler class’

*Fact type:*        Activity has ActivityName

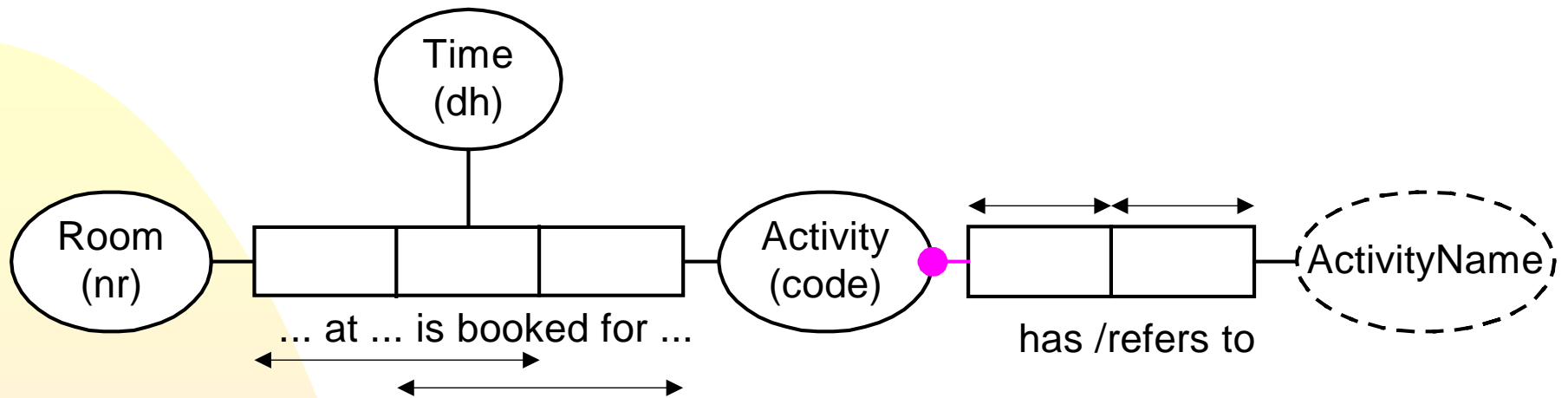
*Predicate:*        ...has ...

# Binary Fact Type



VMC	VisioModeler class
AQD	ActiveQuery demo
SP	Staff Party
...	...

**each** Activity has **exactly one** ActivityName  
**each** ActivityName refers to **at most one** Activity

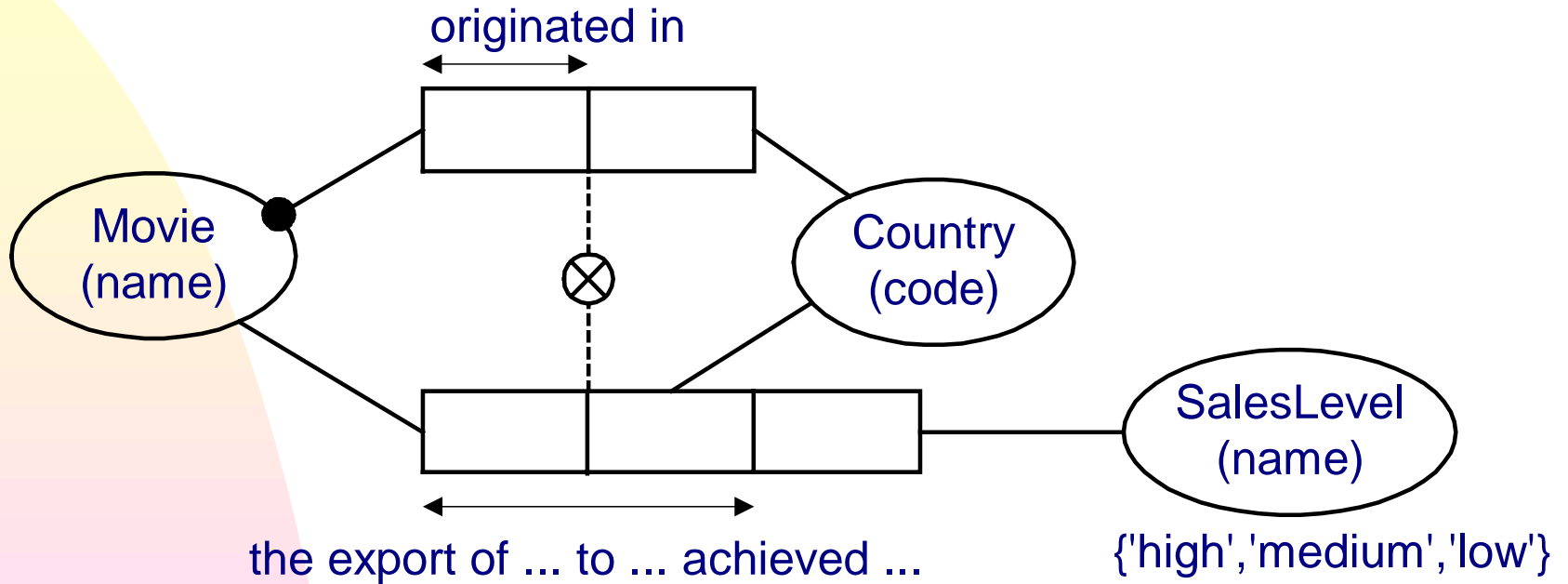


20	Mon 9am	VMC
20	Tue 2pm	VMC
33	Mon 9am	AQD
33	Fri 5pm	SP
...	...	...

AQD	ActiveQuery demo
SP	Staff party
VMC	VisioModeler class
<b>Y2K</b>	<b>Year 2000 seminar</b>
...	...

*Extra data may be needed to illustrate some constraints*

Terminator 2	USA
Crocodile Dundee	Aus
Backdraft	USA

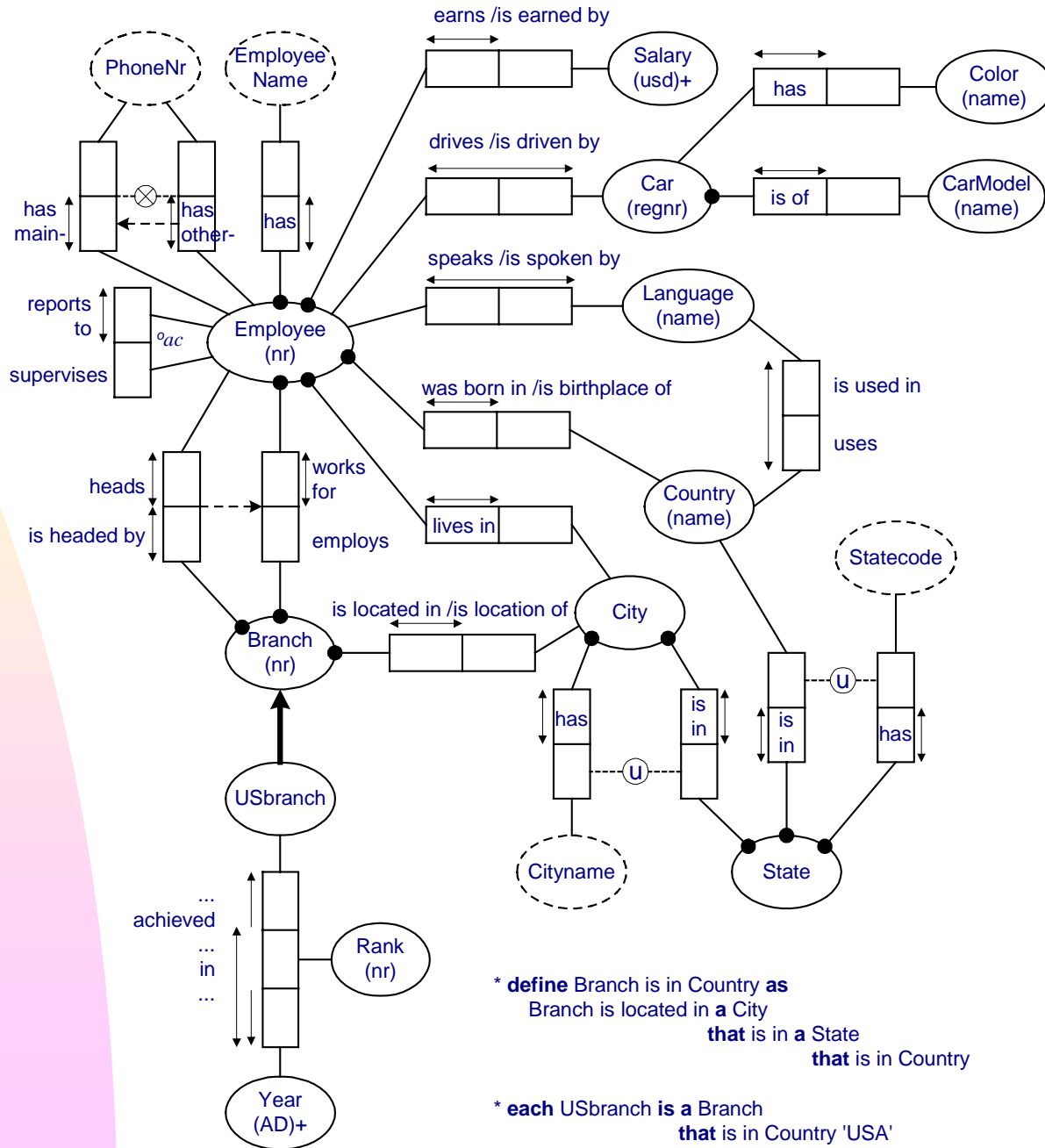


Backdraft	Aus	high
Backdraft	NZ	low
Backdraft	UK	medium
Crocodile Dundee	NZ	low
Crocodile Dundee	USA	high

*connectedness*

*clarity*

*expressibility*



\* **define** Branch is in Country as  
Branch is located in a City  
**that** is in a State  
**that** is in Country

\* **each** USbranch is a Branch  
**that** is in Country 'USA'