

# DATABASE

## INTRODUCTION

Debora Gil, Oriol Ramos, Carles Sanchez

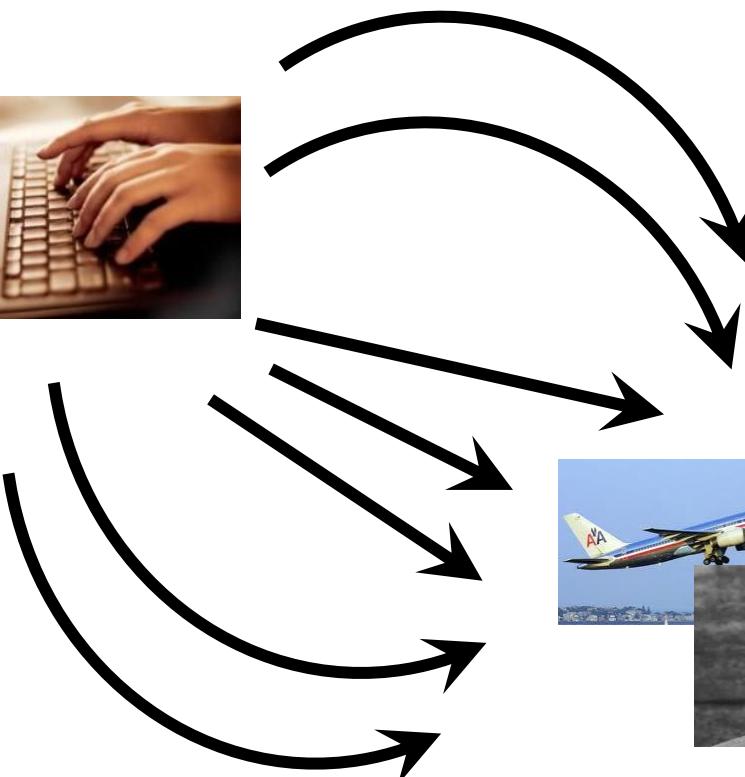
# Outline

1. Basic concepts
2. Database Systems (Dbs)
3. Database components

# 1. Basic Concepts

# Computer Systems

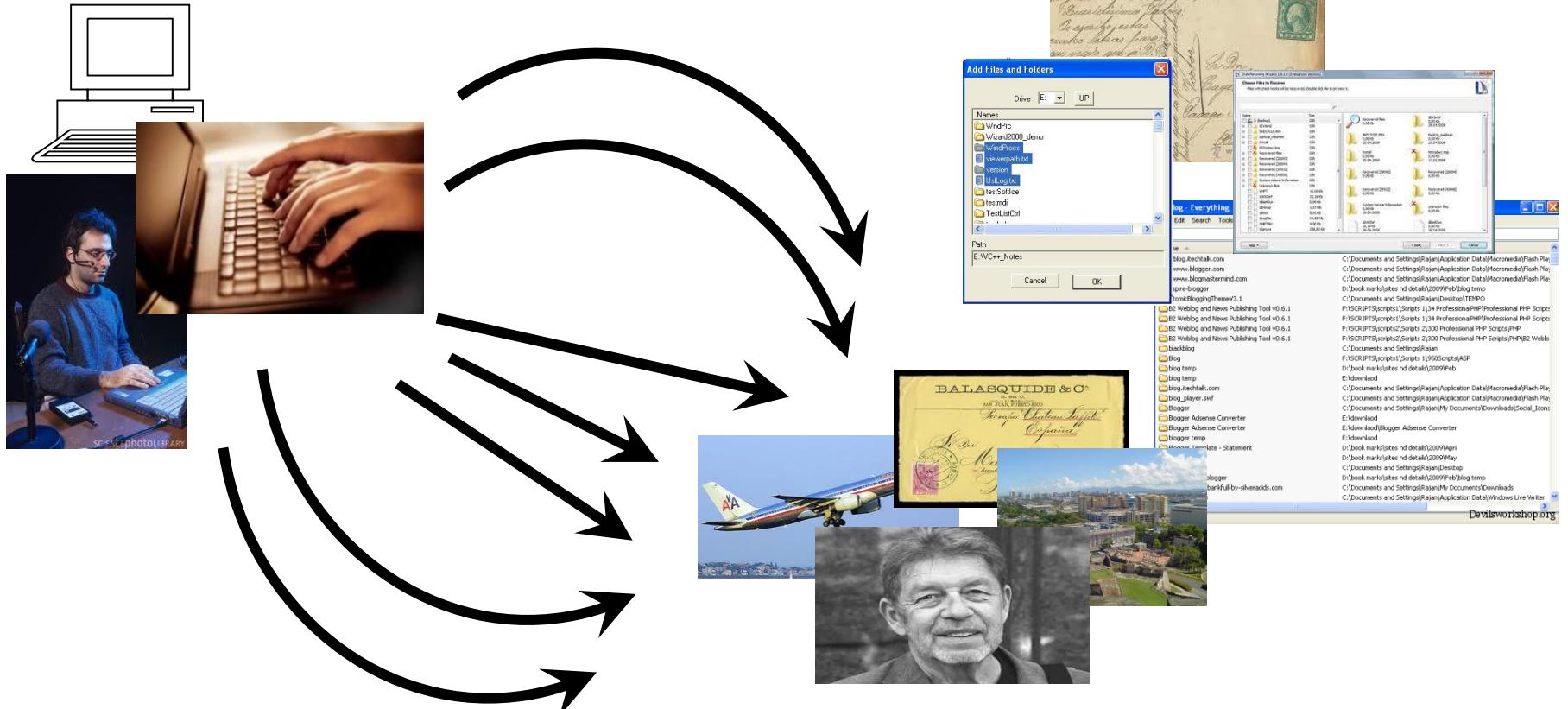
Applications which access to data  
and process them



Stored data

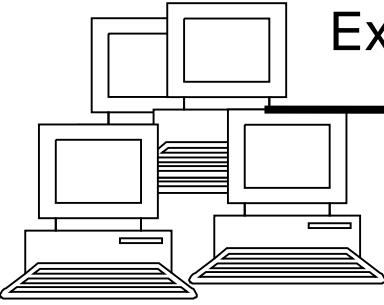
# Engineering Informatics

Software Engineering: Guarantee the functionality of applications



Databases: Way to store data ensuring persistent content and efficient access

# Personal Use



A screenshot of Microsoft Excel showing a data table. The columns include Group, Precio, Total, and others. The data consists of various numerical values and some text entries like "SUSPES".

	Grup	Precio	Total	Autosubsidio	Informatico	Ex. Precio	Total	Autosubsidio	Informatico	Ruta
2	41	6,75	55	4,95	10	4	6	5,5	4,85	5,75
3	41	6,15	7,05	10	5	10	6,15	7,05	6,5	8,44
4	41	5	7,5	10	2	10	5	7,5	6,05	9,98
5	41	1,25	2,2	10	5 SUSPES	2,25	3,4 SUSPES	3,25		3,75
6	41	4,25	3,5	10	2	10	4,25	3,5	3,75	5,42
7	41	7,15	7,25	10	5 *	7,15	7,25	7,15	6,225	8,74
8	41	6,8	5	10	5,85	10	6,8	5	5,3	7,23
9	41	6,75	10	5,85	10	7,2	6,75	6,75	7,25	7,25
10	41	0,05	2,25	NP	4 NP	0,05	2,25 SUSPES	0,05	2,25	1,45
11	41	1	1,25	10	2	2	1,25	0,2 SUSPES		1,25
12	41	6,15	2,5	0,5	7,5	10	6,05	3,5	3,75	7,25
13	41	9,15	8,65	10	5 *	9,15	8,65	9,15	8,25	10,05
14	41	3,5	3,8	9,2	8 *	3,25	3,65	5,85	6,74	8,87
15	41	2,5	1,75	10	6,8 *	1	7,45 SUSPES	1	7,45	4
16	41	6,8	7,25	10	6,8	10	6,8	7,25	7,25	8,27
17	41	7,75	4,5	10	7,1	10	7,75	4,5	6,125	7,67
18	41	5	4,8	9,2	6,6	8	5	4,8	6,0	7,6
19	41	2,2	1,05	10	7,5	10	2,2	1,05 SUSPES	1,05	1,05
20	41	3,65	0,6	7,6 *	8,4	10	3,65	0,6	0,6	8,4
21	41	3,65	0,6	7,6 *	8,4	10	3,65	0,6	0,6	8,4

Name: John

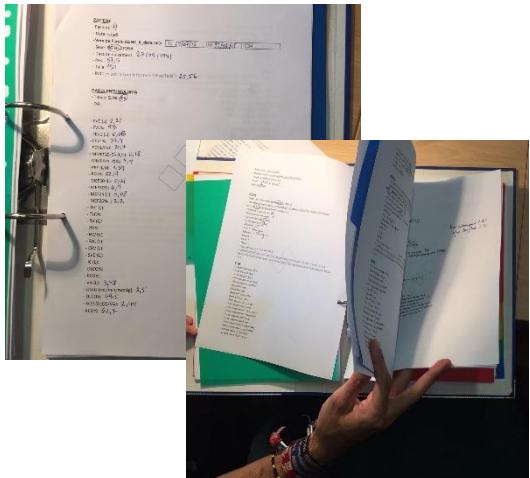
ID: 1234567

Claim number: 32

C ++

Python

Basic



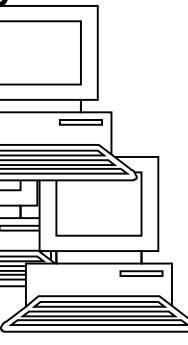
manual

A screenshot of the SPSS Statistics Data Editor. The data table contains variables such as NHC, Sex, Age, and several SUSPES entries. The table has 31 rows and 10 columns.

Name	Type	Width	Decimals	Label	Value	Mining	Columns	Align	Format
1 NHC	Numeric	0	0		None	None	0	Left	Scale
2 Sexo	String	25	0		None	None	25	Left	Text
3 Sexo	Text	2	0	[P, Hombre]	None	None	2	Left	Text
4 Edad	Numeric	2	0		None	None	2	Left	Scale
5 Ctra	Numeric	0	0	[P, Femenino]	None	None	4	Left	Text
6 Edad	Text	2	0	[P, Edad]	None	None	2	Left	Text
7 Diametro	Numeric	0	0		None	None	12	Left	Scale
8 Diametro	Text	2	0	[P, Diametro]	None	None	11	Left	Text
9 Peso	Numeric	0	0		None	None	5	Left	Scale
10 SIGNRON	Numeric	0	0		None	None	7	Right	Text
11 PERCENTA	Numeric	0	0		None	None	0	Left	Scale
12 Unidad	Text	2	0	[P, No]	None	None	0	Left	Text
13 Localización	Numeric	0	0		None	None	7	Left	Scale
14 UMBIL	Numeric	0	0		None	None	11	Left	Scale
15 BRONQU	Numeric	0	0		None	None	0	Left	Text
16 RENAL	Numeric	0	0		None	None	11	Left	Text
17 BAI	Numeric	0	0		None	None	23	Left	Text
18 Raspado	Numeric	0	0		None	None	22	Left	Text
19 Raspado	Text	2	0	[P, Raspado]	None	None	2	Left	Text
20 Biopsia	Numeric	0	0		None	None	13	Left	Text
21 Numb	Numeric	0	0		None	None	0	Right	Text
22 NumbP	Numeric	0	0		None	None	0	Right	Text
23 BAL	Numeric	0	0	[P, No]	None	None	0	Right	Text
24 MECO	Numeric	0	0	[P, No]	None	None	0	Right	Text
25 MECO	Text	2	0	[P, No]	None	None	2	Right	Text
26 TEPIT	Numeric	0	0	[P, No]	None	None	0	Right	Text
27 TEPITTotal	Numeric	0	0	resumen de expresión	None	None	1	Left	Text
28 TEPITTotal	Text	2	0	resumen de expresión	None	None	2	Left	Text
29 RADIC	Numeric	0	1	radiograf	None	None	12	Left	Text
30 RADIC	Text	2	0	[P, Radiograf]	None	None	2	Left	Text
31 DIAGBR	Numeric	0	0		None	None	23	Left	Text

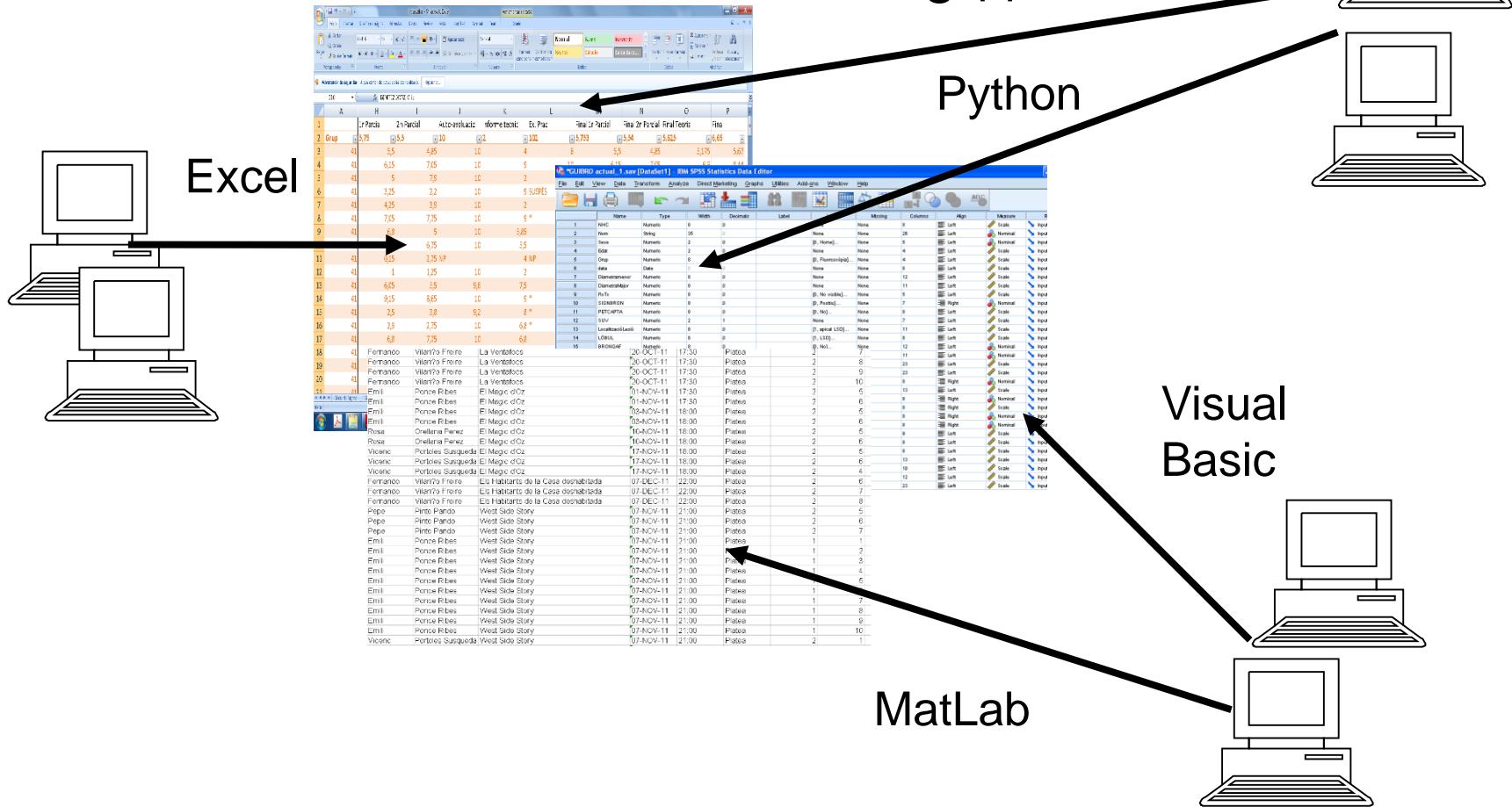
SPSS

MatLab



Each user application has its own system with its particular format and access

# Shared Big Data



There is a **centralized system** with standarized format and access for different applications

# Access to large volumes of modifiable data

What would happen to a standard program using access by file if either one attribute is removed or the order/type of attributes are changed?

DNI	NOM	COGNOMS	ESPECTACLE	DATA	HORA	ZONA	FILA	NUMERO
11111111	Pepe	Pinto Pandc	Entre Tres	27/01/2012	18:00	Platea	1	3
11111111	Pepe	Pinto Pandc	Mar i Cel	02/03/2012	21:00	Platea	1	10
11111111	Pepe	Pinto Pandc	Entre Tres	23/02/2012	23:00	Platea	2	3
11111111	Pepe	Pinto Pandc	Entre Tres	23/02/2012	23:00	Platea	2	4
11111111	Pepe	Pinto Pandc	Entre Tres	01/03/2012	22:00	Platea	2	5
11111111	Pepe	Pinto Pandc	Entre Tres	01/03/2012	22:00	Platea	2	6
11111111	Pepe	Pinto Pandc	Entre Tres	17/03/2012	18:00	Platea	1	5
11111111	Pepe	Pinto Pandc	Entre Tres	17/03/2012	18:00	Platea	1	6
11111111	Pepe	Pinto Pandc	Els Pastorets	01/01/2012	18:00	Pis	1	5
22222222	José	Colorado G	Hamlet	06/04/2012	22:00	Platea	3	4
22222222	José	Colorado G	Hamlet	06/04/2012	22:00	Platea	3	5
22222222	José	Colorado G	Hamlet	06/04/2012	22:00	Platea	3	6
22222222	José	Colorado G	La extraña pareja	08/06/2012	22:00	Platea	1	5
33333333	Fernando	Vilariño Fre	Entre Tres	10/02/2012	18:00	Platea	1	5
33333333	Fernando	Vilariño Fre	Veus búlgares	29/09/2011	21:00	Platea	2	6
33333333	Fernando	Vilariño Fre	L'auca del senyor Esteve	23/02/2012	21:00	Platea	2	1
33333333	Fernando	Vilariño Fre	L'auca del senyor Esteve	23/02/2012	21:00	Platea	2	2
33333333	Fernando	Vilariño Fre	Els Pastorets	12/01/2012	18:00	Platea	1	6
33333333	Fernando	Vilariño Fre	Els Pastorets	12/01/2012	18:00	Platea	1	7
33333333	Fernando	Vilariño Fre	Els Pastorets	12/01/2012	18:00	Platea	1	8
33333333	Fernando	Vilariño Fre	La Ventafo	20/10/2011	17:30	Platea	2	1
33333333	Fernando	Vilariño Fre	La Ventafo	20/10/2011	17:30	Platea	2	2
33333333	Fernando	Vilariño Fre	La Ventafo	20/10/2011	17:30	Platea	2	3
33333333	Fernando	Vilariño Fre	La Ventafo	20/10/2011	17:30	Platea	2	4
33333333	Fernando	Vilariño Fre	La Ventafo	20/10/2011	17:30	Platea	2	5
33333333	Fernando	Vilariño Fre	La Ventafo	20/10/2011	17:30	Platea	2	6
33333333	Fernando	Vilariño Fre	La Ventafo	20/10/2011	17:30	Platea	2	7
33333333	Fernando	Vilariño Fre	La Ventafo	20/10/2011	17:30	Platea	2	8
33333333	Fernando	Vilariño Fre	La Ventafo	20/10/2011	17:30	Platea	2	9
33333333	Fernando	Vilariño Fre	La Ventafo	20/10/2011	17:30	Platea	2	10

# Shared files in centralized system with linked data

File: Tickets.txt

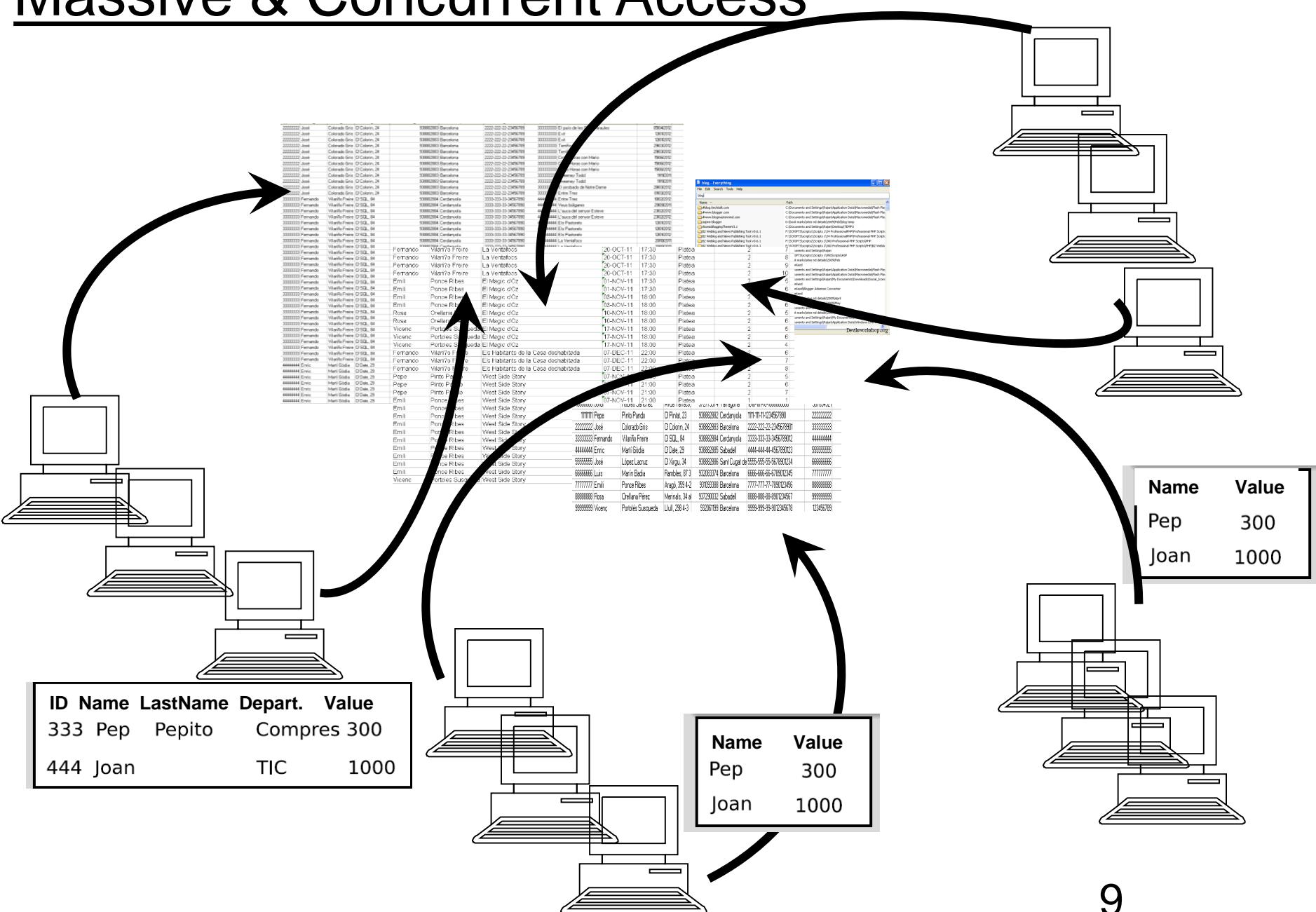
Fernando	Vilariño Freire	La Ventafocs	20-OCT-11	17:30	Platea	2	7
Fernando	Vilariño Freire	La Ventafocs	20-OCT-11	17:30	Platea	2	8
Fernando	Vilariño Freire	La Ventafocs	20-OCT-11	17:30	Platea	2	9
Fernando	Vilariño Freire	La Ventafocs	20-OCT-11	17:30	Platea	2	10
Emili	Ponce Ribes	El Magic d'Oz	01-NOV-11	17:30	Platea	2	5
Emili	Ponce Ribes	El Magic d'Oz	01-NOV-11	17:30	Platea	2	6
Emili	Ponce Ribes	El Magic d'Oz	03-NOV-11	18:00	Platea	2	5
Emili	Ponce Ribes	El Magic d'Oz	03-NOV-11	18:00	Platea	2	6
Rosa	Orellana Perez	El Magic d'Oz	10-NOV-11	18:00	Platea	2	5
Rosa	Orellana Perez	El Magic d'Oz	10-NOV-11	18:00	Platea	2	6
Vicenc	Portoles Susqueda	El Magic d'Oz	17-NOV-11	18:00	Platea	2	5
Vicenc	Portoles Susqueda	El Magic d'Oz	17-NOV-11	18:00	Platea	2	6
Vicenc	Portoles Susqueda	El Magic d'Oz	17-NOV-11	18:00	Platea	2	4
Fernando	Vilariño Freire	Els Habitants de la Casa deshabitada	07-DEC-11	22:00	Platea	2	6
Fernando	Vilariño Freire	Els Habitants de la Casa deshabitada	07-DEC-11	22:00	Platea	2	7
Fernando	Vilariño Freire	Els Habitants de la Casa deshabitada	07-DEC-11	22:00	Platea	2	8
Pepe	Pinto Pando	West Side Story	07-NOV-11	21:00	Platea	2	5
Pepe	Pinto Pando	West Side Story	07-NOV-11	21:00	Platea	2	6
Pepe	Pinto Pando	West Side Story	07-NOV-11	21:00	Platea	2	7

File: Clients.txt

ID	Nom	Cognom	Direcció	Telèfon	Codi Postal	Localitat	CP	CP2
1111111	Pepe	Pinto Pando	C/ Pintat, 23	938882882	Cerdanyola	1111-11-11-1234567890	222222222	
2222222	José	Colorado Gris	C/ Colorin, 24	938882883	Barcelona	2222-222-22-2345678901	333333333	
3333333	Fernando	Vilariño Freire	C/ SQL, 84	938882884	Cerdanyola	3333-333-33-3456789012	444444444	
4444444	Enric	Martí Gòdia	C/ Date, 29	938882885	Sabadell	4444-444-44-4567890123	555555555	
5555555	José	López Lacruz	C/ Xirgu, 34	938882886	Sant Cugat del Vallès	5555-555-55-5678901234	666666666	
6666666	Luis	Marín Badia	Rambles, 87 3	932083374	Barcelona	6666-666-66-6789012345	777777777	
7777777	Emili	Ponce Ribes	Aragó, 359 4-2	931093388	Barcelona	7777-777-77-7890123456	888888888	
8888888	Rosa	Orellana Pérez	Merinals, 34 al	937290032	Sabadell	8888-888-88-8901234567	999999999	
9999999	Vicenc	Portolés Susqueda	Llull, 298 4-3	932061199	Barcelona	9999-999-99-9012345678	123456789	

It is necessary to guarantee a correct propagation of changes  
(consistency of the information)

# Massive & Concurrent Access



# Database systems

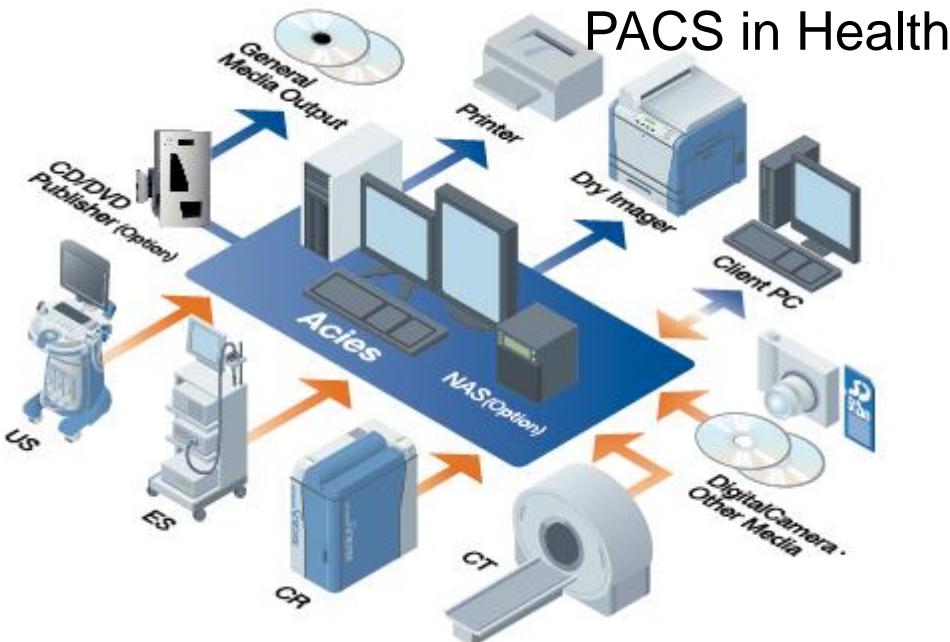
- ✓ Gives efficient access to large data volumes (access time, computer resources)
- ✓ Allow different requirements (format, sorting) of user data (multiuser centralized system)
- ✓ Stores data with modifiable structure

## 2. Database systems (DBS)

**Definition:** Computer storage systems for the manipulation of related data volumes in centralized system (multiuser).

- Insert new data into existing files
- View data content
- Update data (delete, insert)
- Delete and add files
- They allow to guarantee the coherence of the content.
- They describe objects from the real world at different abstraction levels.

# Examples

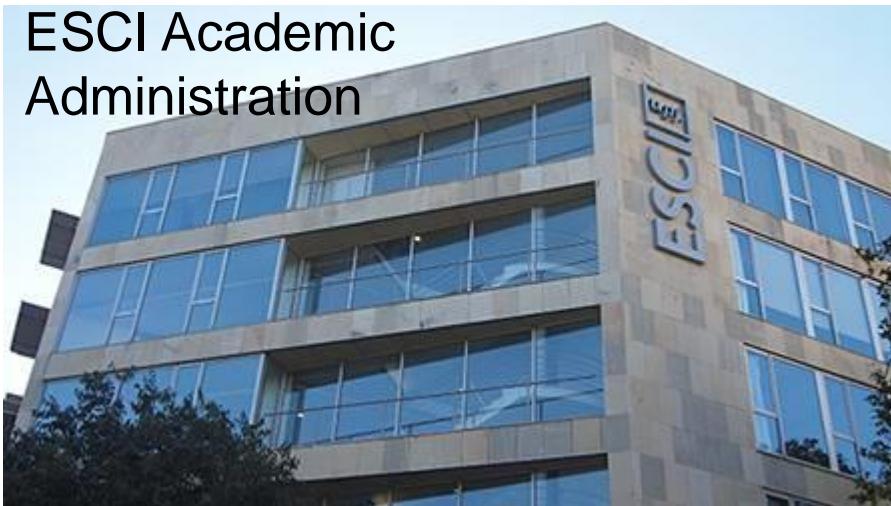


PACS in Health



Banking and accounting systems

ESCI Academic Administration



Apps



# Paradigms of the DBSs

## Relational

Data homogeneity:

Logical structure which always keeps the same type of information (varying on number and type)

Data integrity:

Always guarantee ACID properties (Atomicity, Consistency, Isolation, Durability)

PostgreSQL



ORACLE

## Non-Relational

Data heterogeneity:

Flexible logical structure that allows to keep the same type of information (varying on number and type)

Data integrity:

Always guarantee ACID properties (Atomicity, Consistency, Isolation, Durability)

 mongoDB®

The mongoDB logo consists of a green leaf icon followed by the word "mongoDB" in a grey sans-serif font, with a registered trademark symbol at the end.

 OrientDB®

The OrientDB logo features a circular sunburst icon with orange, yellow, and red segments, followed by the word "OrientDB" in a grey sans-serif font, with a registered trademark symbol at the end.

# DBSs Requirements

In order to guarantee a proper operation of all applications using data stored in a DB, the system must:

- Guarantee the **data integrity**
- Be **efficient**
- Grant **independency** between program code and data format

# Integrity

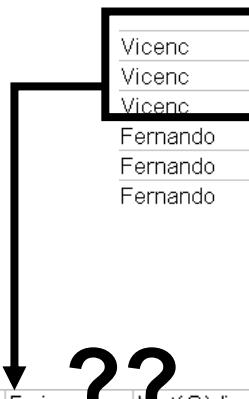
- Content **consistency** and **coherence** of information stored in the DB.
- Avoid **redundancy** which are incorrect or duplicated values in the requested information.
- DBS must guarantee the data **integrity when content is modified** (insert, update and changes propagation).

# Integrity. Incoherence

**Definition:** impossible value of the requested information

Avoid incorrect values in the requested information

Name: Pepe  
**DNI: -22.89792317**  
ClaimNumber: 33  
Value: 0 €  
**Age: 5000**



Vicenc	Portoles Susqueda	El Magic d'Oz	17-NOV-11	1
Vicenc	Portoles Susqueda	El Magic d'Oz	17-NOV-11	1
Vicenc	Portoles Susqueda	El Magic d'Oz	17-NOV-11	1
Fernando	Vilari?o Freire	Els Habitants de la Casa deshabitada	07-DEC-11	2
Fernando	Vilari?o Freire	Els Habitants de la Casa deshabitada	07-DEC-11	2
Fernando	Vilari?o Freire	Els Habitants de la Casa deshabitada	07-DEC-11	2
44444444 Enric	Martí Gòdia	C/ Date, 29	938882885	
55555555 José	López Lacruz	C/ Xirgu, 34	938882886	
66666666 Luis	Marián Badia	Rambles, 87 3	932083374	
77777777 Emili	Ponce Ribes	Aragó, 359 4-2	931093388	
88888888 Rosa	Orellana Pérez	Merinals, 34 al	937290032	

# Integrity. Inconsistency

**Definition:** same query gives different results depending on its implementation.

Two sources of information (files, field's files) which are equivalent (same semantical content) have different values

Name: Pepe  
DNI: 3333333A  
ClaimNumber: 33  
**Antiquity: 10**  
Amount: 100000

Name: Pepe  
DNI: 3333333A  
ClaimNumber: 33  
**Antiquity: 40**  
Amount: 100000

The diagram illustrates a consistency issue between two data sources. On the left, two boxes represent different implementations. The top box shows a record for 'Pepe' with 'Antiquity: 10'. The bottom box shows a record for 'Pepe' with 'Antiquity: 40'. Arrows point from these records to two separate tables on the right. The top table, associated with antiquity 10, lists 'Emili' multiple times with 'West Side Story' and '07-NOV-11'. The bottom table, associated with antiquity 40, lists 'Emilia' once with 'West Side Story' and '07-NOV-11'. This visualizes how the same query can yield different results based on the specific implementation.

Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Vicenc	Portoles Susqueda	West Side Story	07-NOV-11

22222222	José	Colorado Gris	C/ Colorin, 24	938882
33333333	Fernando	Vilaríñ Freire	C/ SQL, 84	938882
44444444	Enric	Martí Gòdia	C/ Date, 29	938882
55555555	José	López Lacruz	C/ Xirgu, 34	938882
66666666	Luis	Marcia Radio	Rambles, 87 3	932083
77777777	Emilia	Ponce Ribes	Aragó, 359 4-2	931093
88888888	Rosa	Tirallons Pérez	Marimbla, 24 s/n	937299

# Integrity. Redundancy

Repeated information has the following problems:

Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Emili	Ponce Ribes	West Side Story	07-NOV-11
Vicent	Portoles Susqueda	West Side Story	07-NOV-11

44444444	Enric	Martí Gòdia	Ci Date, 29	938882
55555555	José	López Lacruz	Ci Xirgu, 34	938882
66666666	Luis	Marín Badia	Rambles, 87 3	932083
77777777	Emilia	Ponce Ribes	Aragó, 359 4-2	931093
88888888	Rosa	Iriellana Pérez	Merinals 34 al	937291

- Unnecessary disk usage
- Duplication on updates
- Risk of inconsistencies

DBS must guarantee minimum redundancy and update on information to all files if one is changed (change propagation)

# Integrity. Examples

- Marks in negative → Incoherence
- More than two practical marks for the same student → Inconsistency

	A	B	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Grup	NIA	1r Parcial	2n Parcial	Auto-avaluació	Informe tecnic	Ex. Prac	Final 1r Parcial	Final 2n Parcial	Final Teori	Condicions	NP	Promig brut	Final Expedient	Aprovats
2	41	1423875	4,75	3,25	10,00	5,80	5,00	4,75	7,75	6,25	1,00	1,00	6,16	6,2	1,00
3	41	1423991	7,65	6,35	10,00	5,50	10,00	7,65	6,35	7,00	1,00	1,00	7,90	7,9	1,00
4	41	1424840	1,75	0,00	7,14	3,30	4,00	1,75	0,00	*	SUSPÈS	1,00	2,92	2,9	0,00
5	41	1325938	6,40	5,65	10,00	9,10	6,00	6,40	5,65	6,03	1,00	1,00	7,03	7,0	1,00
6	41	1457898	5,25	4,50	0,00	2,50	5,00	5,25	6,90	6,08	1,00	1,00	4,43	4,4	0,00
7	41	1428718	3,00	NP	5,00	0,00	0,00	4,70	2,50	*	SUSPÈS	1,00	1,94	1,9	0,00
8	41	1423177	7,50	8,65	10,00	8,50	10,00	7,50	8,65	8,08	1,00	1,00	8,93	8,9	1,00
9	41	1423722	8,50	8,50	10,00	5,80	7,00	8,50	8,50	8,50	1,00	1,00	7,66	7,7	1,00
10	41	1359074	1,15	4,00	9,05	4,00	5,00	5,30	4,00	4,65	SUSPÈS	1,00	5,06	4,5	0,00
11	41	1397752	3,25	2,50	10,00	3,30	4,00	3,25	2,50	*	SUSPÈS	1,00	4,01	4,0	0,00
12	41	1423762	6,55	7,25	10,00	5,80	10,00	6,55	7,25	6,90	1,00	1,00	7,92	7,9	1,00
13	41	1424454	3,25	NP	0,00	5,30	10,00	5,85	4,10	5,00	1,00	1,00	6,05	6,1	1,00
14	41	1424822	7,50	6,40	8,57	8,50	7,00	7,50	6,40	6,95	1,00	1,00	7,44	7,4	1,00
15	41	1425085	4,85	7,65	10,00	5,80	8,00	7,25	7,65	7,45	1,00	1,00	7,54	7,5	1,00
16	41	1423802	8,00	8,90	10,00	5,80	10,00	8,00	8,90	8,45	1,00	1,00	8,54	8,5	1,00
17	41	1423574	NP	NP	3,10	0,00	8,00	5,25	6,25	5,75	1,00	1,00	5,01	5,0	1,00
18	41	1391463	8,75	9,75	10,00	4,80	10,00	8,75	9,75	9,25	1,00	1,00	8,66	8,7	1,00
19	41	1423957	7,50	6,20	9,05	6,50	6,00	7,50	6,20	6,85	1,00	1,00	6,74	6,7	1,00
20	41	1425096	2,25	NP	10,00	5,70	10,00	5,20	6,50	5,85	1,00	1,00	7,48	7,5	1,00
21	41	1424801	7,50	9,50	10,00	8,50	5,50	7,50	9,50	8,50	1,00	1,00	7,75	7,8	1,00
22	41	1362742	3,75	3,50	10,00	5,70	9,00	6,05	6,55	6,30	1,00	1,00	7,36	7,4	1,00
23	41	1395060	6,50	6,50	10,00	5,50	7,00	6,50	6,50	6,50	1,00	1,00	6,80	6,8	1,00
24	41	1425111	6,25	6,60	10,00	3,30	9,00	6,25	6,60	6,43	1,00	1,00	6,93	6,9	1,00
25	41	1424348	3,25	NP	10,00	5,70	7,00	6,25	5,75	6,00	1,00	1,00	6,64	6,6	1,00
26	41	1325626	8,40	6,65	10,00	8,50	10,00	8,40	6,65	7,53	1,00	1,00	8,71	8,7	1,00
27	41	1424116	8,15	8,10	10,00	4,20	7,00	8,15	8,10	8,13	1,00	1,00	7,19	7,2	1,00
28	41	1391319	5,75	5,00	10,00	5,50	8,00	5,75	5,00	5,38	1,00	1,00	6,65	6,7	1,00
29	41	1412688	4,00	4,25	9,29	0,00	6,00	4,85	4,25	4,55	SUSPÈS	1,00	4,55	4,5	0,00
30	41	1391967	NP	NP	9,52	3,70	5,00	6,30	2,25	*	SUSPÈS	1,00	4,90	4,5	0,00
31	41	1335425	2,00	3,00	8,33	0,00	9,00	3,50	5,75	*	SUSPÈS	1,00	5,38	4,5	0,00
32	41	1330701	4,05	5,00	2,38	6,00	6,00	4,05	7,16	5,61	1,00	1,00	5,48	5,5	1,00
33	41	1391630	6,75	6,50	2,14	5,70	5,00	6,75	6,50	6,63	1,00	1,00	5,50	5,5	1,00
34	41	1390533	2,50	2,65	5,48	5,70	5,00	3,00	3,75	*	SUSPÈS	1,00	4,54	4,5	0,00
35	41	1424269	9,00	9,75	10,00	8,50	10,00	9,00	9,75	9,38	1,00	1,00	9,45	9,5	1,00
36	41	1391873	6,25	4,60	10,00	5,80	8,00	6,25	4,60	5,43	1,00	1,00	6,73	6,7	1,00
37	41	1424310	4,75	4,85	10,00	3,30	7,00	4,75	9,15	6,95	1,00	1,00	6,54	6,5	1,00
38	41	1390042	4,75	4,50	10,00	5,70	9,00	4,75	5,15	5,00	1,00	1,00	6,82	6,8	1,00
39	41	1391015	5,75	6,90	10,00	3,90	7,00	5,75	6,90	6,33	1,00	1,00	6,41	6,4	1,00
40	41	1423737	5,75	5,75	10,00	4,80	8,00	5,75	5,75	5,75	1,00	1,00	6,66	6,7	1,00

# Integrity. Examples

Repeated information → Redundancy

Solution: visitor name and surname could be represented as DNI

Fernando	Vilari?o Freire	La Ventafocs	20-OCT-11	17:30	Platea	2	7
Fernando	Vilari?o Freire	La Ventafocs	20-OCT-11	17:30	Platea	2	8
Fernando	Vilari?o Freire	La Ventafocs	20-OCT-11	17:30	Platea	2	9
Fernando	Vilari?o Freire	La Ventafocs	20-OCT-11	17:30	Platea	2	10
Emili	Ponce Ribes	El Magic d'Oz	01-NOV-11	17:30	Platea	2	5
Emili	Ponce Ribes	El Magic d'Oz	01-NOV-11	17:30	Platea	2	6
Emili	Ponce Ribes	El Magic d'Oz	03-NOV-11	18:00	Platea	2	5
Emili	Ponce Ribes	El Magic d'Oz	03-NOV-11	18:00	Platea	2	6
Rosa	Orellana Perez	El Magic d'Oz	10-NOV-11	18:00	Platea	2	5
Rosa	Orellana Perez	El Magic d'Oz	10-NOV-11	18:00	Platea	2	6
Vicenc	Portoles Susqueda	El Magic d'Oz	17-NOV-11	18:00	Platea	2	5
Vicenc	Portoles Susqueda	El Magic d'Oz	17-NOV-11	18:00	Platea	2	6
Vicenc	Portoles Susqueda	El Magic d'Oz	17-NOV-11	18:00	Platea	2	4
Fernando	Vilari?o Freire	Els Habitants de la Casa deshabitada	07-DEC-11	22:00	Platea	2	6
Fernando	Vilari?o Freire	Els Habitants de la Casa deshabitada	07-DEC-11	22:00	Platea	2	7
Fernando	Vilari?o Freire	Els Habitants de la Casa deshabitada	07-DEC-11	22:00	Platea	2	8
Pepe	Pinto Pando	West Side Story	07-NOV-11	21:00	Platea	2	5
Pepe	Pinto Pando	West Side Story	07-NOV-11	21:00	Platea	2	6
Pepe	Pinto Pando	West Side Story	07-NOV-11	21:00	Platea	2	7
Emili	Ponce Ribes	West Side Story	07-NOV-11	21:00	Platea	1	1
Emili	Ponce Ribes	West Side Story	07-NOV-11	21:00	Platea	1	2
Emili	Ponce Ribes	West Side Story	07-NOV-11	21:00	Platea	1	3
Emili	Ponce Ribes	West Side Story	07-NOV-11	21:00	Platea	1	4
Emili	Ponce Ribes	West Side Story	07-NOV-11	21:00	Platea	1	5
Emili	Ponce Ribes	West Side Story	07-NOV-11	21:00	Platea	1	6
Emili	Ponce Ribes	West Side Story	07-NOV-11	21:00	Platea	1	7
Emili	Ponce Ribes	West Side Story	07-NOV-11	21:00	Platea	1	8
Emili	Ponce Ribes	West Side Story	07-NOV-11	21:00	Platea	1	9
Emili	Ponce Ribes	West Side Story	07-NOV-11	21:00	Platea	1	10
Vicenc	Portoles Susqueda	West Side Story	07-NOV-11	21:00	Platea	2	1

# Integrity. Examples

The DBS must guarantee the data integrity when the content is modified:

- Inserts and updates must be adjusted to the defined domain (Coherence)
- Propagation of updates and changes (Consistency)

Final Marks

	A	B	E	F	G	H	I	J	K
1	Grup	NIA	1r Parcial	2n Parcial	Auto-avaliaci	Informe tecnic	Ex. Prac	Final 1r	Final 2n
2	41	1423875	4,75	3,25	10,00	5,80	5,00	4,75	7,75
3	41	1423991	7,65	6,35	10,00	5,50	10,00	7,65	6,35
4	41	1424840	1,75	0,00	7,14	3,30	4,00	1,75	0,00
5	41	1325938	6,40	5,65	10,00	9,10	6,00	6,40	5,65
6	41	1457898	5,25	4,50	0,00	2,50	5,00	5,25	6,90
7	41	1428718	3,00	NP	5,00	0,00	0,00	4,70	2,50
8	41	1423177	7,50	8,65	10,00	8,50	10,00	7,50	8,65
9	41	1423722	8,50	8,50	10,00	5,80	7,00	8,50	8,50
10	41	1359074	1,15	4,00	9,05	4,00	5,00	5,30	4,00
11	41	1397752	3,25	2,50	10,00	3,30	4,00	3,25	2,50
12	41	1423762	6,55	7,25	10,00	5,80	10,00	6,55	7,25
13	41	1424454	3,25	NP	0,00	5,30	10,00	5,85	4,10
14	41	1424822	7,50	6,40	8,57	8,50	7,00	7,50	6,40
15	41	1425085	4,85	7,65	10,00	5,80	8,00	7,25	7,65
16	41	1423802	8,00	8,90	10,00	5,80	10,00	8,00	8,90
17	41	1423574	NP	NP	3,10	0,00	8,00	5,25	6,25
18	41	1391463	8,75	9,75	10,00	4,80	10,00	8,75	9,75
19	41	1423957	7,50	6,20	9,05	6,50	6,00	7,50	6,20
20	41	1425096	2,25	NP	10,00	5,70	10,00	5,20	6,50
21	41	1424801	7,50	9,50	10,00	8,50	5,50	7,50	9,50

1st Partial

NIA	1erParcial
1423875	4,75
1423991	7,65
1424840	1,75
1325938	6,4
1457898	5,25
1428718	3
1423177	7,5
1423722	8,5
1359074	1,15

2on Partial

NIA	Zon Pard
1423875	3,25
1423991	6,35
1424840	0,00
1325938	5,65
1457898	4,50
1428718	NP
1423177	8,65
1423722	8,50

# Efficiency

DBS should minimize the maximum time for a query (given for access to all records) based on contents and functional needs of the DB:

- Storage Structures. Restructuring the disk information distribution (minimize access to disk)
- Compression Techniques. DBS compress the information to be stored into disk (minimize content)
- Characteristics of the devices where the data is stored.
- Concurrency Control of accesses of several users (priorities)

# Independence. Modifiable structure

What would happens if:

Add or remove fields?  
Change the type of fields?

A	B	E	F	G	H	I	J	K	L	M	N	O	P	Q	
1	Grup	NIA	1r Parcial	2n Parcial	Auto-avaluació	Informe tecnic	Ex. Prac	Final 1r Parcial	Final 2n Parcial	Final Teori	Condicions	NP	Promig brut	Final Expedient	Aprovats
2	41	1423875	4,75	3,25	10,00	5,80	5,00	4,75	7,75	6,25	1,00	1,00	6,16	6,2	1,00
3	41	1423991	7,65	6,35	10,00	5,50	10,00	7,65	6,35	7,00	1,00	1,00	7,90	7,9	1,00
4	41	1424840	1,75	0,00	7,14	3,30	4,00	1,75	0,00	*	SUSPÈS	1,00	2,92	2,9	0,00
5	41	1325938	6,40	5,65	10,00	9,10	6,00	6,40	5,65	6,03	1,00	1,00	7,03	7,0	1,00
6	41	1457898	5,25	4,50	0,00	2,50	5,00	5,25	6,90	6,08	1,00	1,00	4,43	4,4	0,00
7	41	1428718	3,00	NP	5,00	0,00	0,00	4,70	2,50	*	SUSPÈS	1,00	1,94	1,9	0,00
8	41	1423177	7,50	8,65	10,00	8,50	10,00	7,50	8,65	8,08	1,00	1,00	8,93	8,9	1,00
9	41	1423722	8,50	8,50	10,00	5,80	7,00	8,50	8,50	8,50	1,00	1,00	7,66	7,7	1,00
10	41	1359074	1,15	4,00	9,05	4,00	5,00	5,30	4,00	4,65	SUSPÈS	1,00	5,06	4,5	0,00
11	41	1397752	3,25	2,50	10,00	3,30	4,00	3,25	2,50	*	SUSPÈS	1,00	4,01	4,0	0,00
12	41	1423762	6,55	7,25	10,00	5,80	10,00	6,55	7,25	6,90	1,00	1,00	7,92	7,9	1,00
13	41	1424454	3,25	NP	0,00	5,30	10,00	5,85	4,10	5,00	1,00	1,00	6,05	6,1	1,00
14	41	1424822	7,50	6,40	8,57	8,50	7,00	7,50	6,40	6,95	1,00	1,00	7,44	7,4	1,00
15	41	1425085	4,85	7,65	10,00	5,80	8,00	7,25	7,65	7,45	1,00	1,00	7,54	7,5	1,00
16	41	1423802	8,00	8,90	10,00	5,80	10,00	8,00	8,90	8,45	1,00	1,00	8,54	8,5	1,00
17	41	1423574	NP	NP	3,10	0,00	8,00	5,25	6,25	5,75	1,00	1,00	5,01	5,0	1,00
18	41	1391463	8,75	9,75	10,00	4,80	10,00	8,75	9,75	9,25	1,00	1,00	8,66	8,7	1,00
19	41	1423957	7,50	6,20	9,05	6,50	6,00	7,50	6,20	6,85	1,00	1,00	6,74	6,7	1,00
20	41	1425096	2,25	NP	10,00	5,70	10,00	5,20	6,50	5,85	1,00	1,00	7,48	7,5	1,00
21	41	1424801	7,50	9,50	10,00	8,50	5,50	7,50	9,50	8,50	1,00	1,00	7,75	7,8	1,00
22	41	1362742	3,75	3,50	10,00	5,70	9,00	6,05	6,55	6,30	1,00	1,00	7,36	7,4	1,00
23	41	1395060	6,50	6,50	10,00	5,50	7,00	6,50	6,50	6,50	1,00	1,00	6,80	6,8	1,00
24	41	1425111	6,25	6,60	10,00	3,30	9,00	6,25	6,60	6,43	1,00	1,00	6,93	6,9	1,00
25	41	1424348	3,25	NP	10,00	5,70	7,00	6,25	5,75	6,00	1,00	1,00	6,64	6,6	1,00
26	41	1325626	8,40	6,65	10,00	8,50	10,00	8,40	6,65	7,53	1,00	1,00	8,71	8,7	1,00
27	41	1424116	8,15	8,10	10,00	4,20	7,00	8,15	8,10	8,13	1,00	1,00	7,19	7,2	1,00
28	41	1391319	5,75	5,00	10,00	5,50	8,00	5,75	5,00	5,38	1,00	1,00	6,65	6,7	1,00
29	41	1412688	4,00	4,25	9,29	0,00	6,00	4,85	4,25	4,55	SUSPÈS	1,00	4,55	4,5	0,00
30	41	1391967	NP	NP	9,52	3,70	5,00	6,30	2,25	*	SUSPÈS	1,00	4,90	4,5	0,00
31	41	1335425	2,00	3,00	8,33	0,00	9,00	3,50	5,75	*	SUSPÈS	1,00	5,38	4,5	0,00
32	41	1330701	4,05	5,00	2,38	6,00	6,00	4,05	7,16	5,61	1,00	1,00	5,48	5,5	1,00
33	41	1391630	6,75	6,50	2,14	5,70	5,00	6,75	6,50	6,63	1,00	1,00	5,50	5,5	1,00
34	41	1390533	2,50	2,65	5,48	5,70	5,00	3,00	3,75	*	SUSPÈS	1,00	4,54	4,5	0,00
35	41	1424269	9,00	9,75	10,00	8,50	10,00	9,00	9,75	9,38	1,00	1,00	9,45	9,5	1,00
36	41	1391873	6,25	4,60	10,00	5,80	8,00	6,25	4,60	5,43	1,00	1,00	6,73	6,7	1,00
37	41	1424310	4,75	4,85	10,00	3,30	7,00	4,75	9,15	6,95	1,00	1,00	6,54	6,5	1,00
38	41	1390042	4,75	4,50	10,00	5,70	9,00	4,75	5,15	5,00	1,00	1,00	6,82	6,8	1,00
39	41	1391015	5,75	6,90	10,00	3,90	7,00	5,75	6,90	6,33	1,00	1,00	6,41	6,4	1,00
40	41	1423737	5,75	5,75	10,00	4,80	8,00	5,75	5,75	5,75	1,00	1,00	6,66	6,7	1,00

# Independence. Access by content

-Independence between physical and logical.

marks1.dat

Name: Pepe  
DNI: 3333333  
ClaimNumber: 33  
Paid: 600

-Table: Object (logical) from which you access to the data (physical)

Name	NIA	ClaimNum	Paid
Pepe	3333333	33	600
Joan	3333336	34	0

marks2.dat

Name: Joan  
DNI: 3333336  
ClaimNumber: 34  
Paid: 0

Rows: Tuples or records (“individuals”, “objects”)

Columns: Attributes or fields (“properties”)

Logical

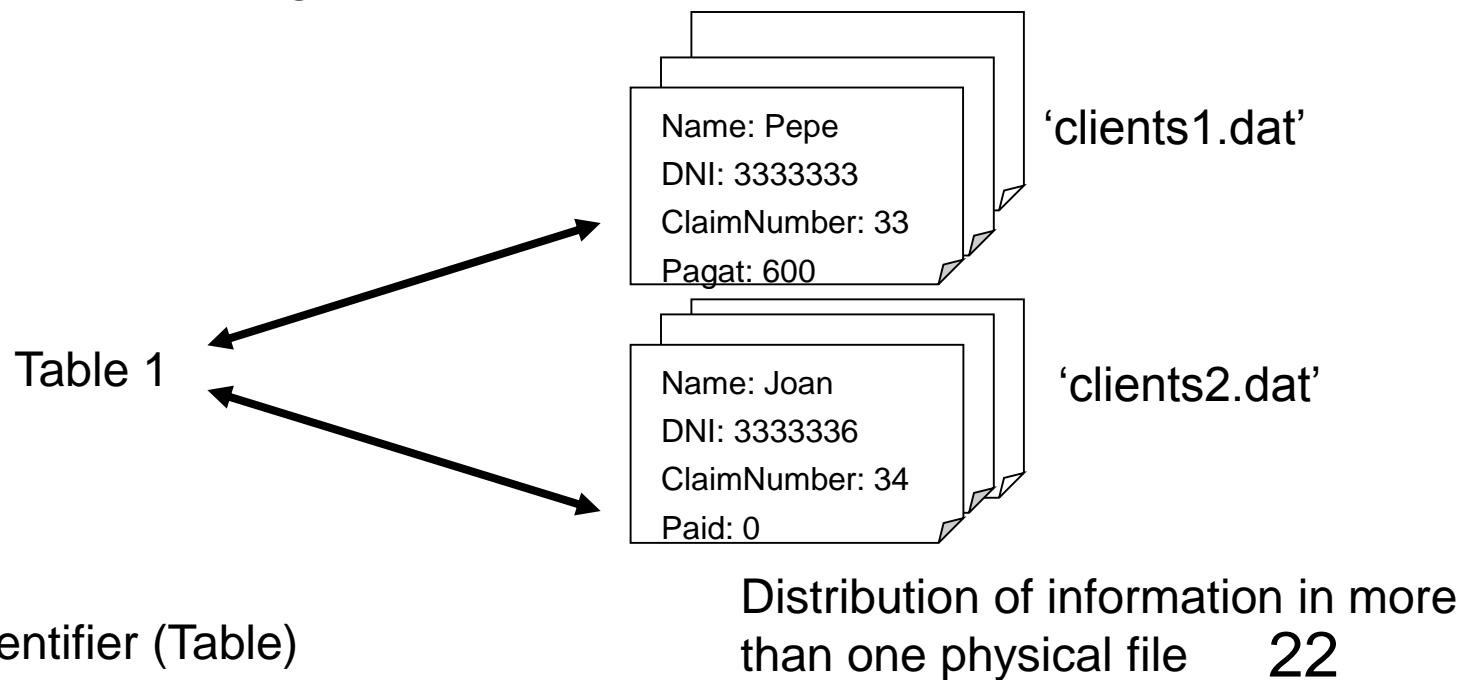
Physical

# Independence. Access by content

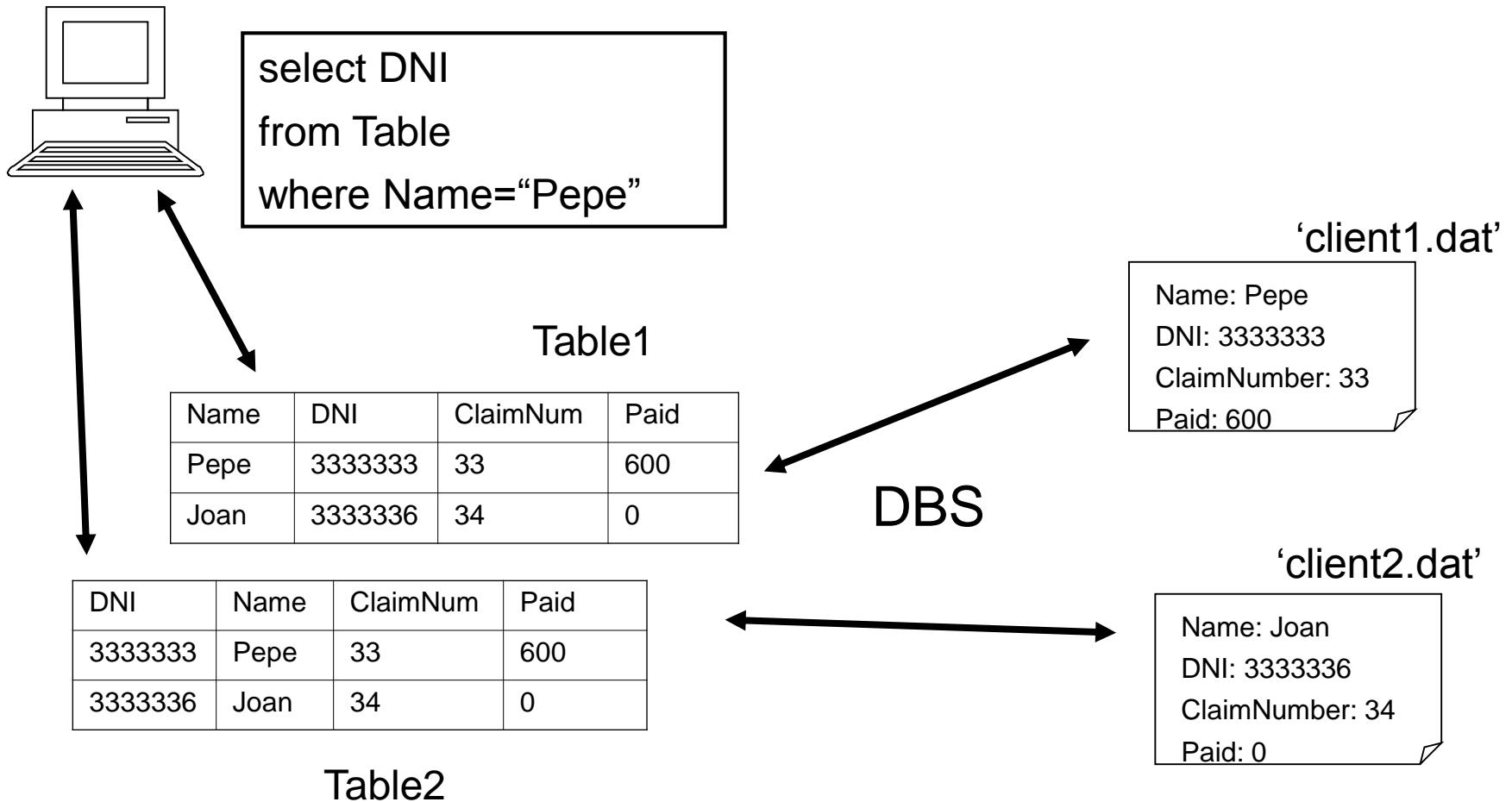
DBS must ensure that changes in the storage structure and data access **do not affect** the applications that use them

## Main requirement of a DBS

DBS guarantees simultaneous access to more than 1 physical file with the same logical file (identifier)



# Independence. Access by content



Logical access is by content and the DBS is responsible for searching the information at its corresponding physical address.

### 3.Components of a Database System

# Components of a DBS

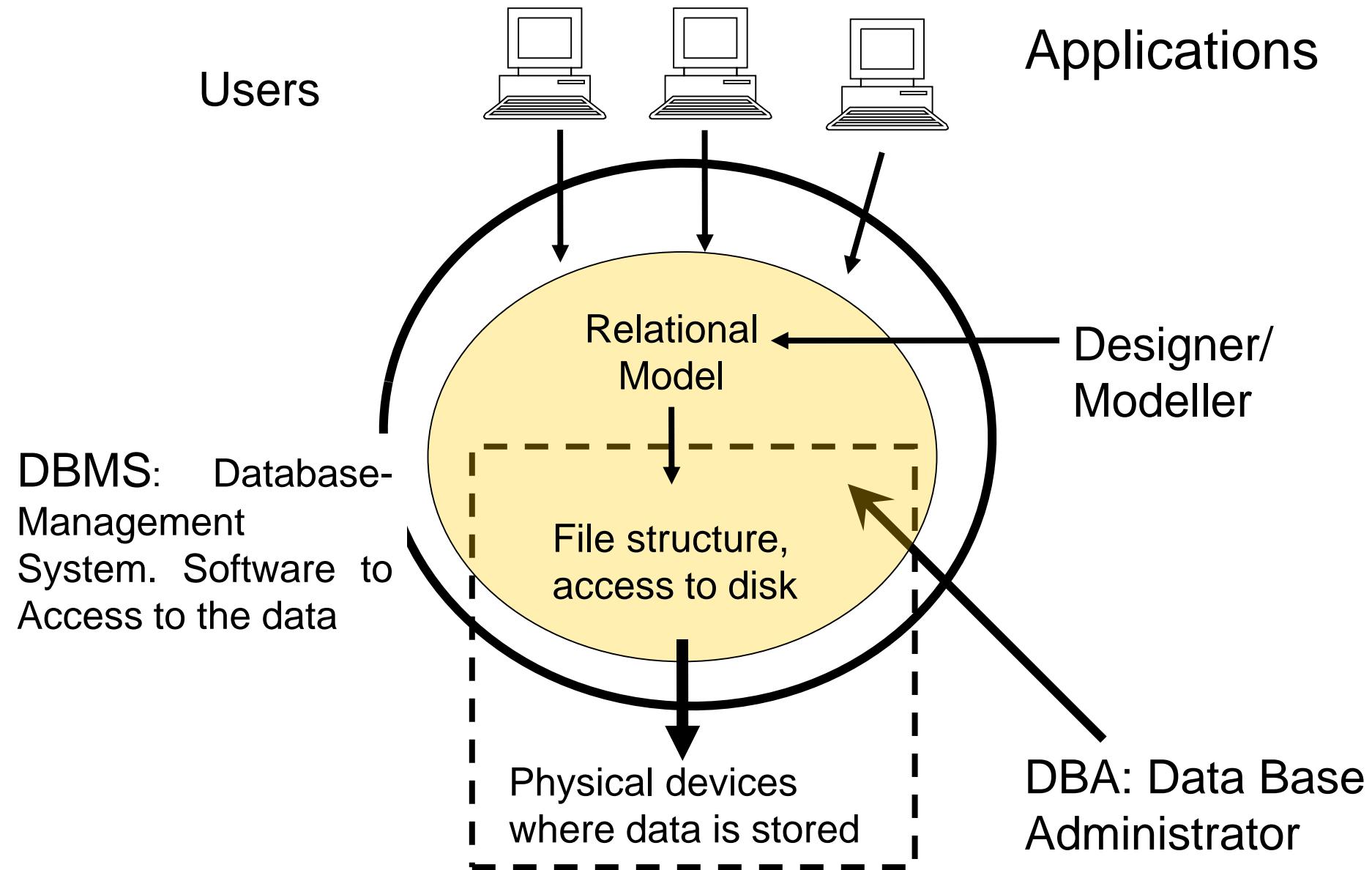
Computers

- Databases
- Hardware (where DBS is installed)
- Database-Management System (DBMS)
- Applications

Humans (Actors)

- Modelers
- DB Administrators DBA
- Programmers
- Final Users

# Component: Users



# Component: Users

**Final user.** Access to the DBS using applications developed by programmers

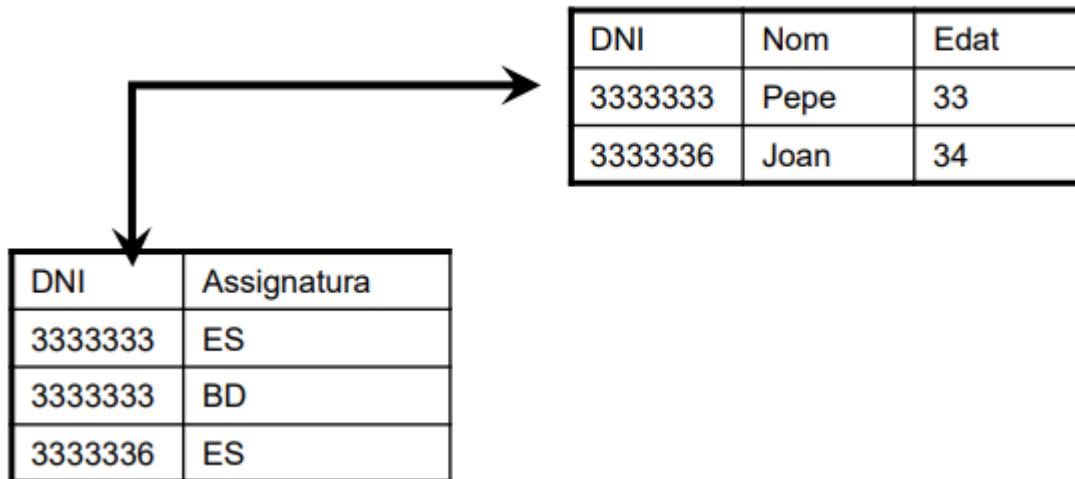
**Programmer.** Design applications that perform queries to the DBS using the DBMS language.

**Data Base Administrator (DBA).** Responsible to maintenance of the DBS:

- File system (physical design)
- Permissions for data access
- Updating of information
- Security backups
- Performance system

# Component: Database

Description of the data in related tables (Relational model) with access by content regardless of the structure of file which save the data and the type of disk access

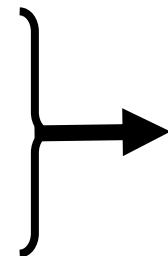


# Component: Hardware

Disks (save data)

RAM (save data & execute software)

CPU (execute software)



Capacity and  
access time are  
important for the  
DBS performance

Administrator manages the selection of the hardware

# Component: DBMS

Data Base Management System (DBMS)

Software which allows create, keep and access to a DB



PostgreSQL



# Component: DBMS

Manage access to DBS:

- Creation of the DB
- User requests (insert, consult, delete, ...)
- Write data to the files
- Data protection and access permissions
- Provides an independent view of hardware operating system through tables, attributes and a handling data language (SQL – Structured Query Language)

# Component: DBMS

Manage and guarantee many properties of a DBS:

- Data integrity (propagation of changes)
- Access Control (security)
- Security backups
- Management of multiple user interfaces
- Performance of a DB (distribution of data in file)

# DATABASE

# ARCHITECTURE

Debora Gil, Oriol Ramos, Carles Sanchez

# Outline

1. Definition
2. ANSI/SPARC
3. Client-Server

# Definition

Structure and description of DBS in modules to guarantee the independence

Most common architectures:

ANSI/SPARC

Client-Server:

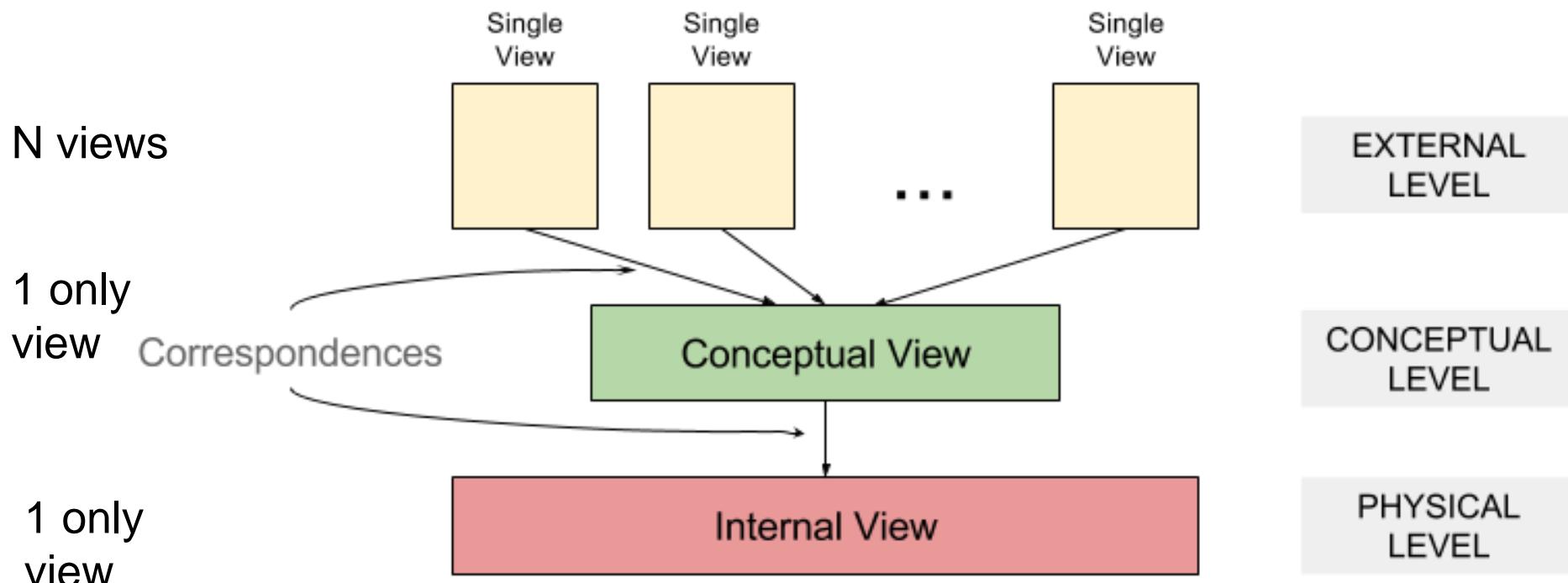
Back-end/Front-end

Distributed systems

## 2. ANSI/SPARC

# ANSI/SPARC

Structure the DB according to the description of the data in 3 levels of abstraction:

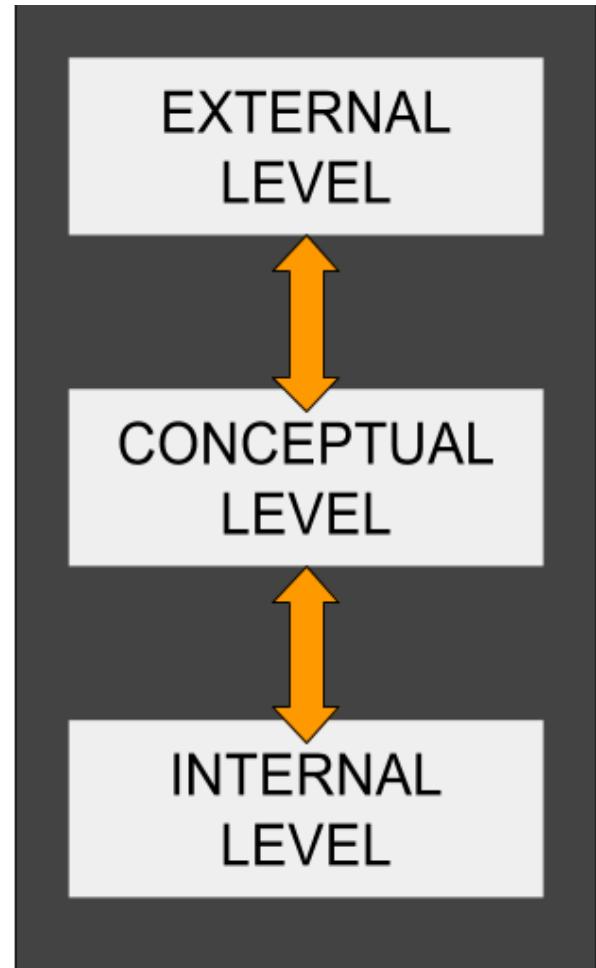


# ANSI/SPARC. Abstraction Levels

External Level. Presentation of data to users (applications that use a relational model).

Conceptual Level. Logical description of data: tables (relational), collections (non-relational)

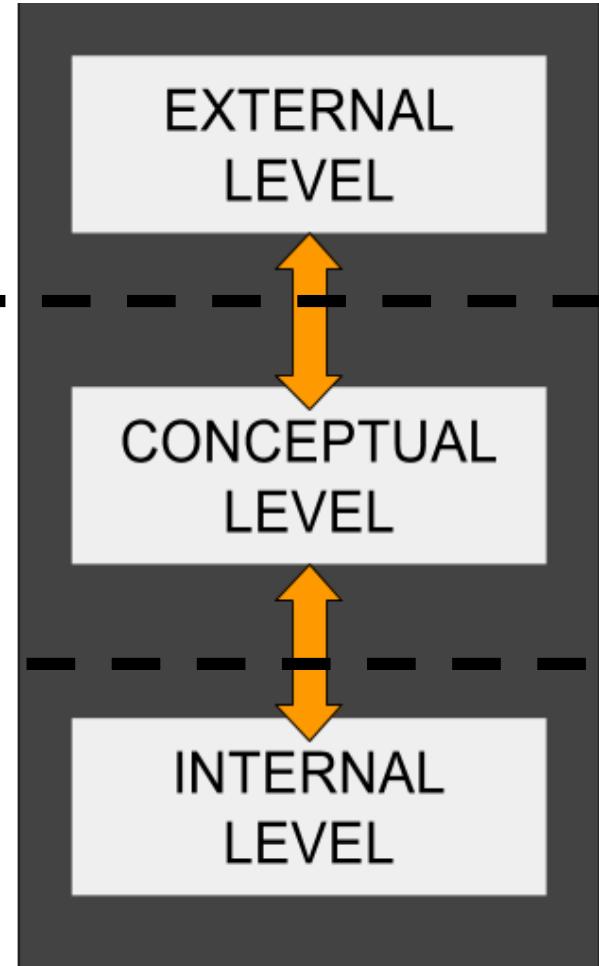
Internal Level. Organization and storage on physical files (low-level, pointers, indexes, ...)



# Data Independence

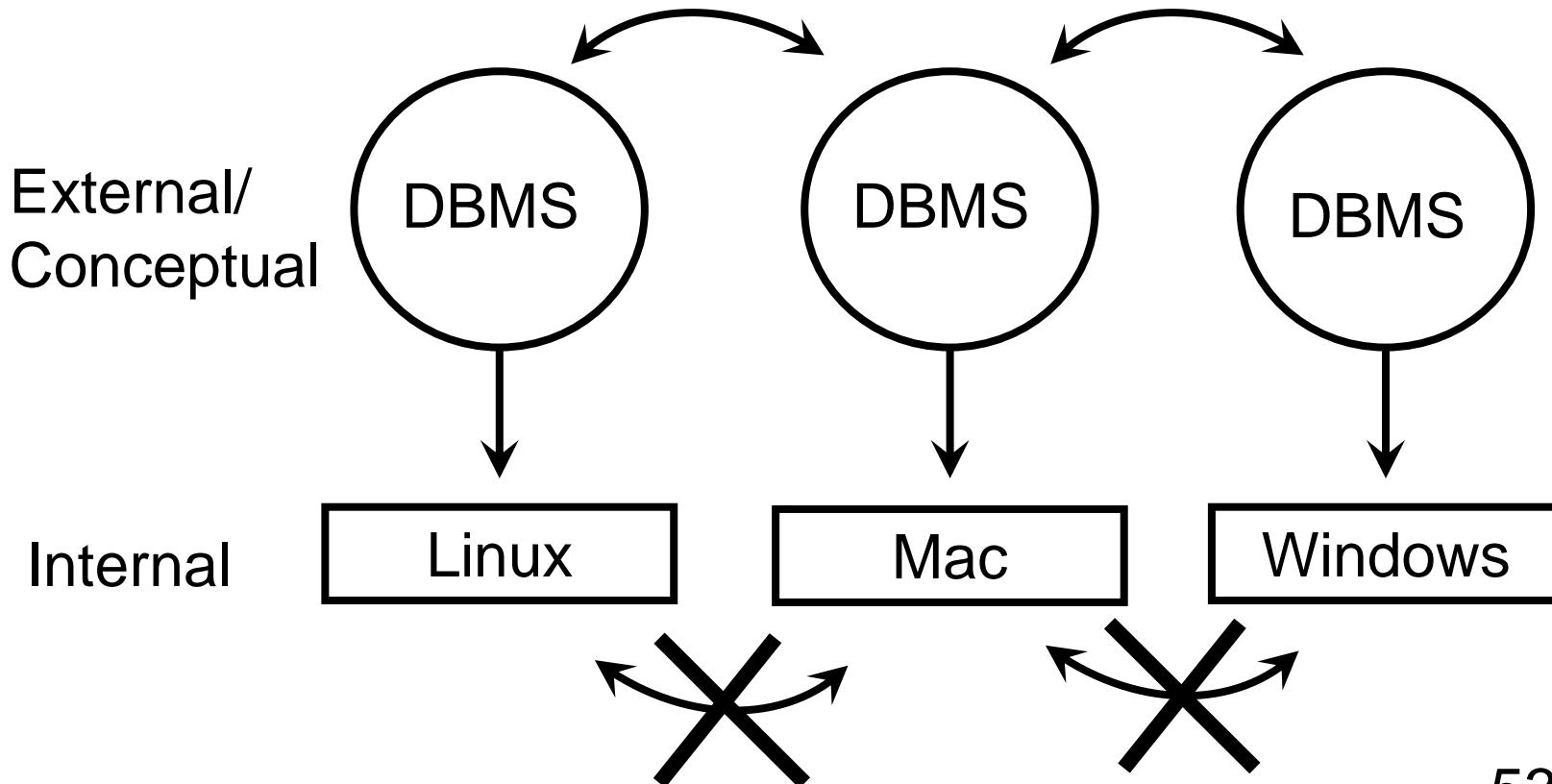
Logic: Modification of the conceptual level without affecting the external level

Physical: Internal distribution does not affect the functionalities of other levels



# Data Independence

External and Conceptual Levels **do not** depend of the operating system. Internal level **does** depends of the operating system.

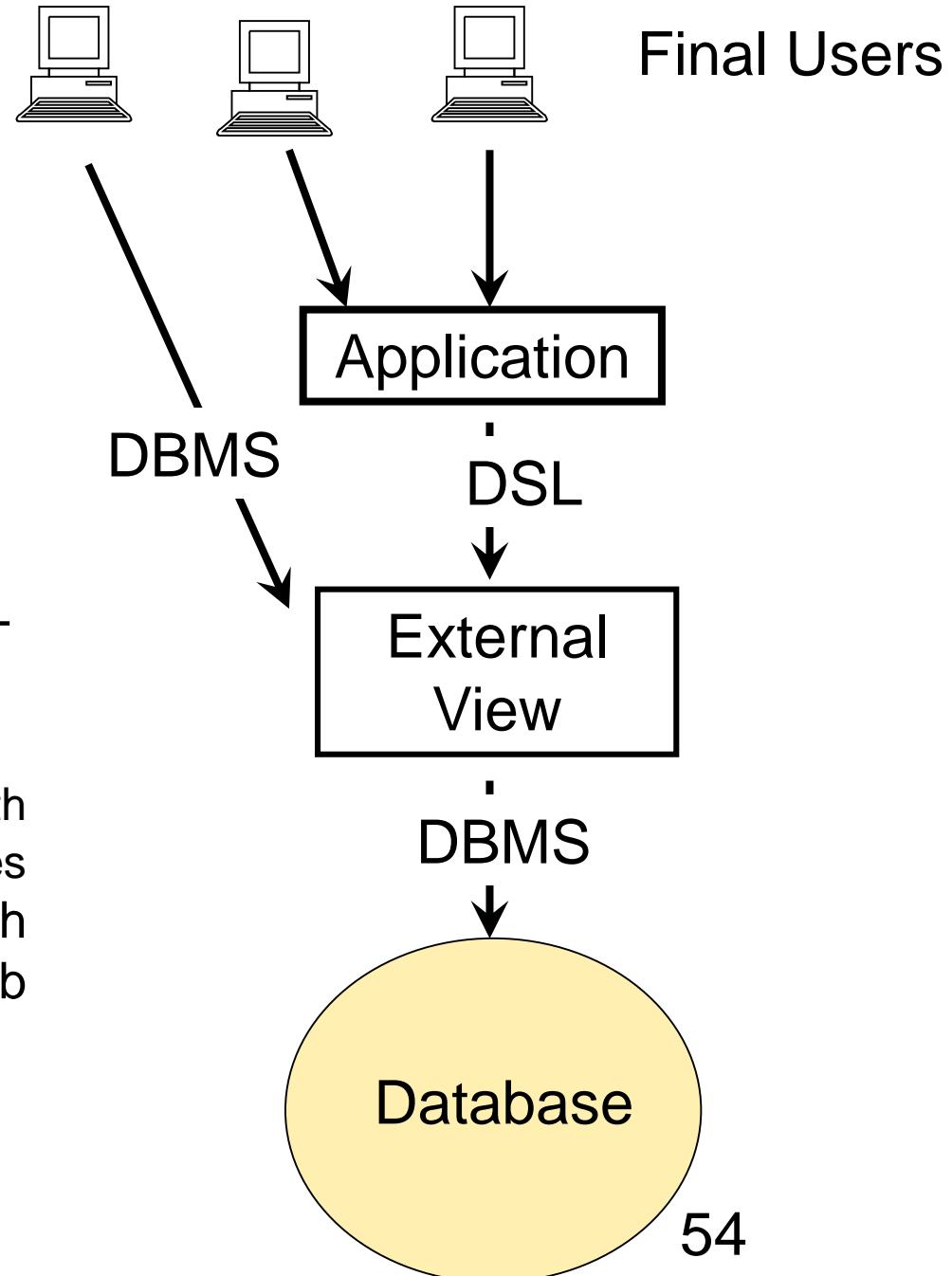


# External Level

Presentation of the data to users.

Final user. Accessing via SQL or some application

Application. Developed with high-end programming languages (C++, PASCAL,...) which incorporate a Data Sub Language (DSL).



# External Level

View: Content (attributes values) on the part of the DB as seen by a user/application at a given time

Name	Paid	€
Pep	300	
Joan	1000	



DSL sub-language in the high-end language is able for interaction with the DBS: Data Definition Language, Data Manipulation Language

DNI	Name	LastName	Dep.	Paid
333	Pep	Pepito	Compres	300
444	Joan		TIC	1000



Name	Paid	\$
Pep	395	
Joan	1.317	

# Conceptual Level

View: Content (attribute values) of **ALL** the **DB** at any given time

DNI	Name	Last Name	Dep.	Paid
333	Pep	Pepito	Compres	300
444	Joan		TIC	1000

Defined based on DDL conceptual (script SQL:  
create table, create domain, create foreign key, ...)

Specify integrity and security controls

There must be a correspondence between the outer and the conceptual scheme

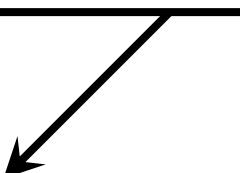
# Internal Level

Content of the files which store the content of tables

Does not match with conceptual view (data distributed in different files)

Files description according the operating system format

DNI	Name	ClaimNum
3333333	Pepe	33
3333336	Joan	34



Filename, table's attributes, access (indexing)

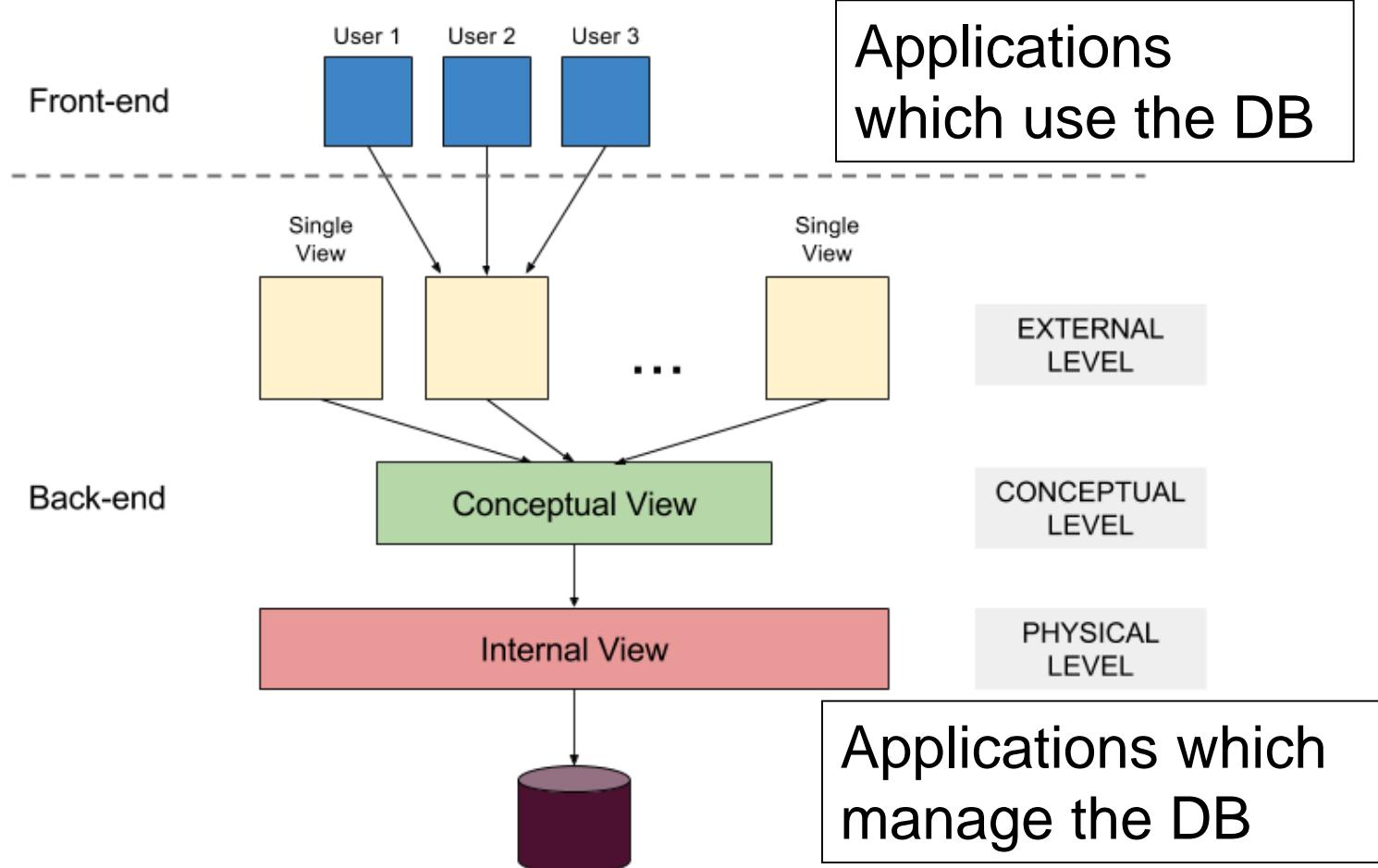
### 3. Client-Server

1 Back-end/Front-end

2 Distributed Systems

# Front-end/Back-end

Structuring the DB according the type of application (software) that are executed on the DB in 2 levels:



# Front-end / Back-end: Components

BACK-END (behind). Software which executes all functions specified in a DBMS



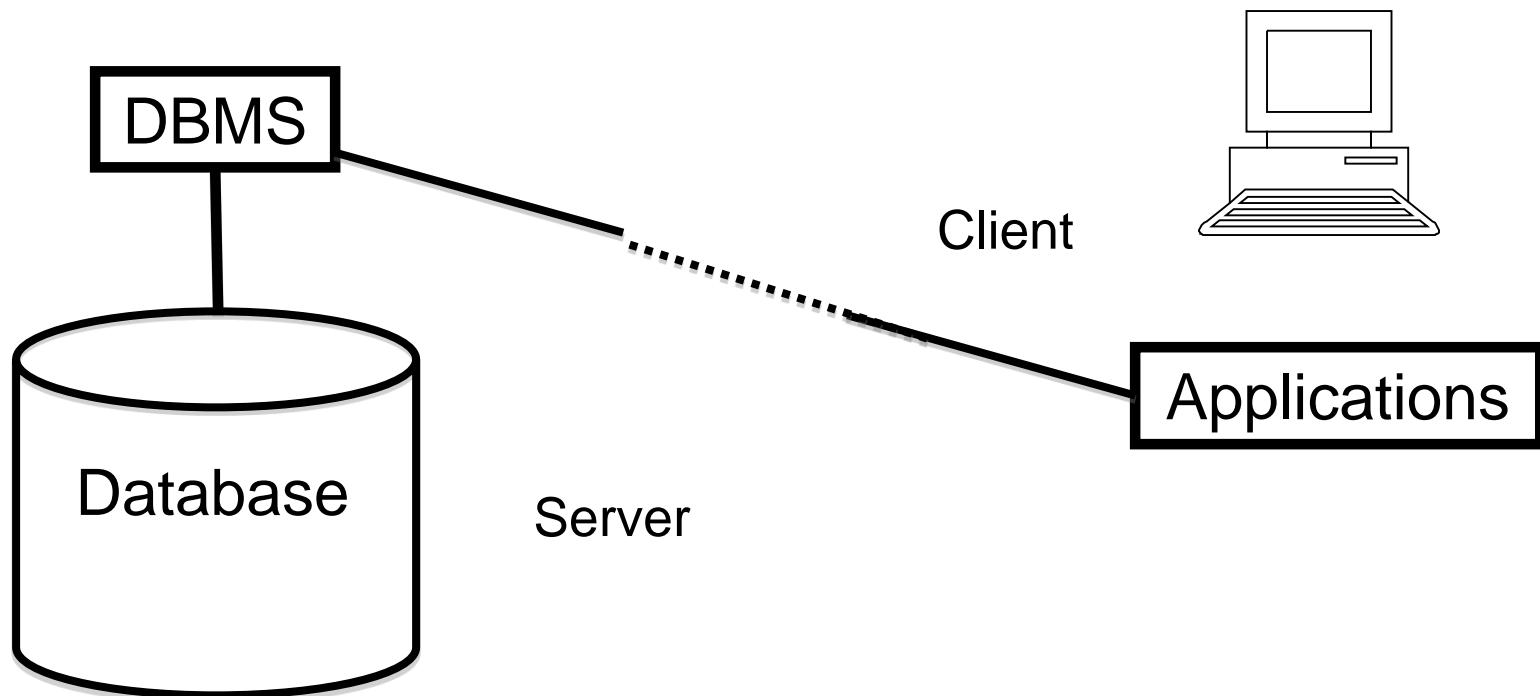
FRONT-END (on front). Applications executed over a DBMS (web browser, smartphone app, SQL developer, etc.)

Usually **1** BACK-END and  
**n** FRONT-ENDs



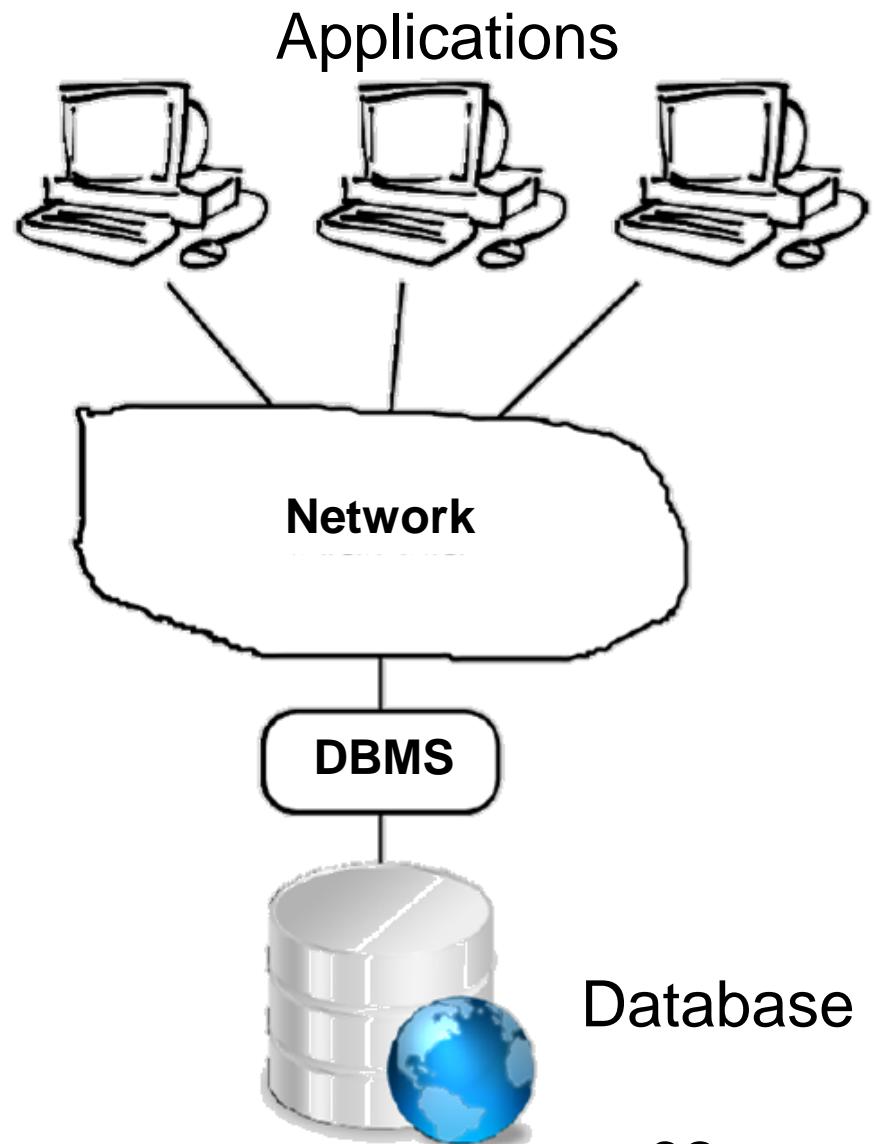
# Distributed System

The separation of the DB according to the software allows each part to be in different physical nodes (or computers).



# Distributed System: Client-Server

**Client (FRONT-END):**  
Machine which executes  
the application



**Server (BACK-END):** Machine  
which executes the DBMS.

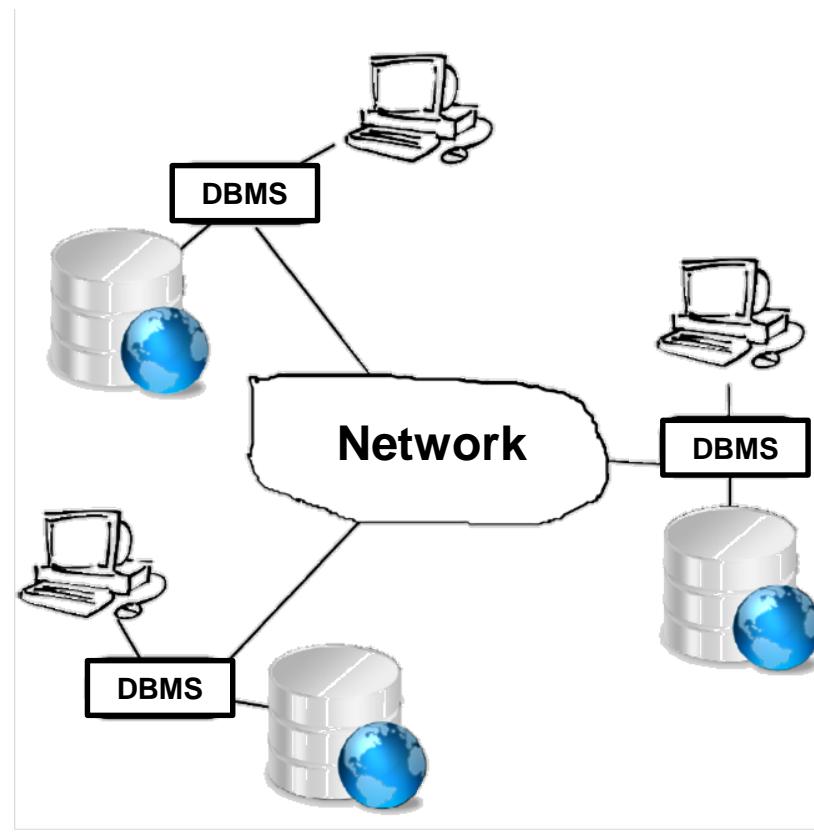
# Distributed System on Network

Distribution of data on several machines and operating systems.

N client-server locals (nodes) are communicating between them using SQL .

Each machine contains a DBS that works like:

- Server per certain users.
- Client for other users.



# Block 2

# Basic ER-Design (PART1)

Debora Gil, Oriol Ramos, Carles Sánchez, Alejandro Párraga

# Basic ER Design Contents

1. E-R Model Introduction

2. Basic Structures

    2.1 Entities

    2.2 Attributes

    2.3 Relationships

3. Relationships Features

    3.1 Cardinality

    3.2 Degree

    3.3 Participation

# Basic ER Design Contents

1. E-R Model Introduction

2. Basic Structures

    2.1 Entities

    2.2 Attributes

    2.3 Relationships

3. Relationships Features

    3.1 Cardinality

    3.2 Degree

    3.3 Participation

# 1. E-R Model Introduction

# Definition

Description / Schematic representation of a real situation (which our application must manage) that requires saving information from different objects (entities) that are related to each other.

Example:

**University Enrolment Management:** In an enrolment management system I want to know which students have enrolled in engineering subjects.

# Definition

Description / Schematic representation of a real situation (which our application must manage) that requires saving information from different objects (entities) that are related to each other.

Exemple:

**Enrolment Management:** In an enrolment management system I want to know which students have enrolled in engineering subjects.

For each student (and subject) I want to keep certain information (name, contact details, ...). In particular, I will have to save the necessary data so as not to confuse 2 students

# Components

They let you describe WHO DOES WHAT:

**Entities.** Object (event) from real world → “Subjects, complements”

**Relationships** (Relations, Interrelations). Association relating several entities → “Verbs”

**Attributes.** Entities and relationships Information or Features

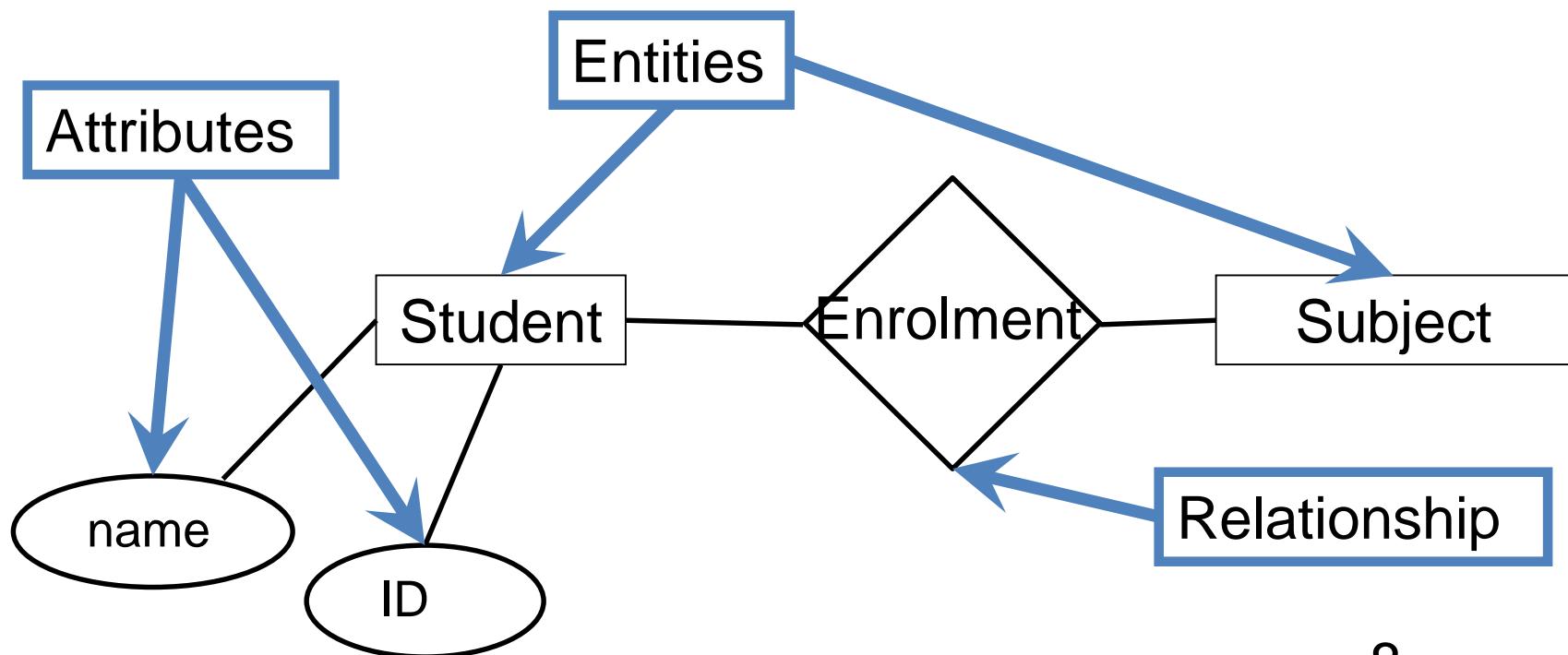
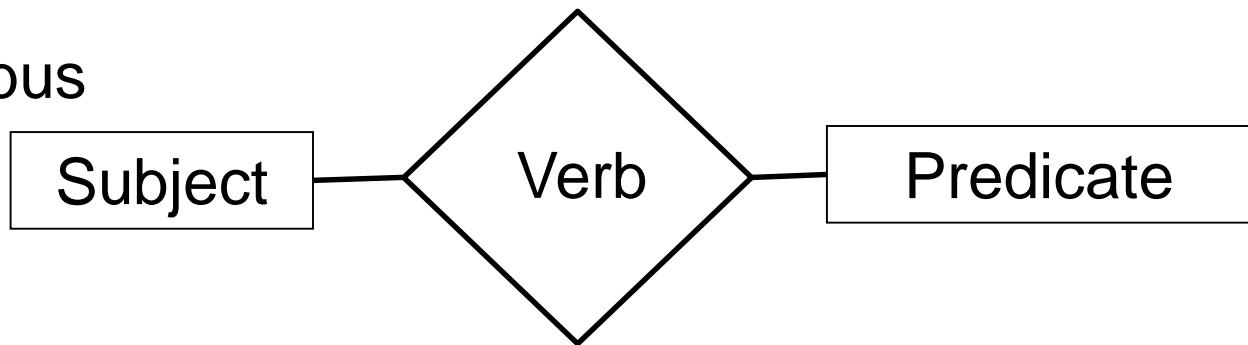
**E-R Diagram** is a graphical representation of all entities and relationships with their attributes

# E-R Diagram

Relationships → rhombus

Attributes → ellipses

Entities → rectangles



## 2. Basic Structures

2.1 Entities

2.2 Attributes

## 2.1 Entities

2.1.1 Definition

2.1.2 Candidate Keys

2.1.3 Primary Key

# Definition

Real-world object or action distinguishable from the rest of which we are interested in keeping some properties. Semantic description of an object. They are represented by squares.

An entity can be Concrete (corresponds to a physical object) or Abstract (corresponds to an action or concept).

Entities have a set of characteristics (attributes) with values that uniquely identify each instance (case, tuple) of the entity

# Examples

- A university **student** is a concrete entity.
- Each student must have a characteristic that uniquely identifies them. For example, NIU 1007899 could uniquely identify a particular student.
- Similarly, book loans can be considered as abstract entities, and the UAB library's loan code L-15AJY9 uniquely identifies "loan" instances.

# Candidate Keys

Every entity must have a subset of its attributes with values that uniquely identify each instance (case, tuple) of the entity:

**Candidate Keys (CK):**

Minimum set of attributes that uniquely identify each instance. A set of attributes that uniquely identify an instance and do not contain other keys inside or redundant attributes

# Candidate Keys

**Identify:** CK attribute values are not repeated. Different instances have some of the CK attributes different

**Minimality:** If we drop any of CK attributes we could have instances with the same values of CK and therefore they would no longer identify

# Examples

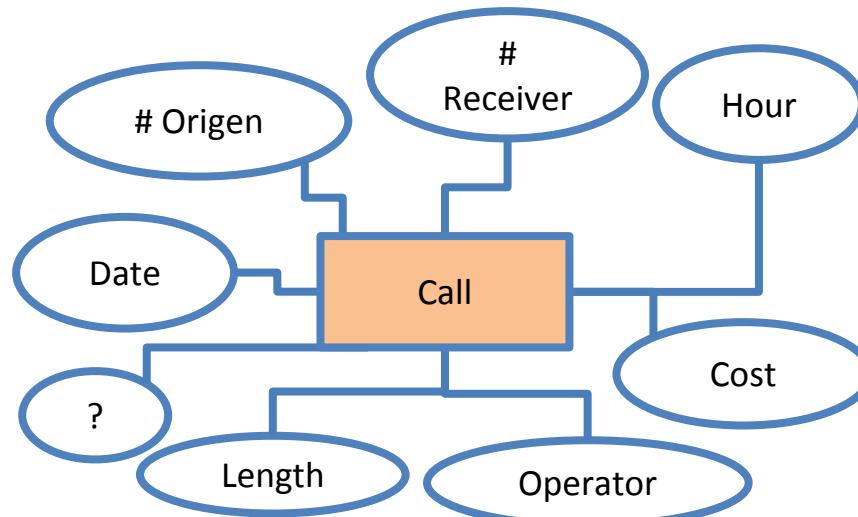
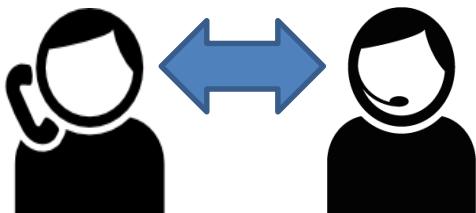
What are the necessary attributes to uniquely identify...

- A member?



- Name
- Surname
- Date of birth
- DNI
- Marital status
- Telephone
- Member code
- Address

- a phone call?



# Examples

- Member

1. DNI

2. Member Code

DNI		Date	Member Code	Name
3676373L		1-1-22	123	Juan
4748474P		1-1-22	333	Juan
6727271Q		1-1-23	667	Pere



Equivalents as a minimal identifiers

#Origin	Cost	Hour	Date	#Receiver	Min
66234560	0.8	10:00	1-9-23	93581444	1
93581444	0.4	10:20	1-9-23	66234501	5
66234501	0.8	10:00	2-9-23	93581444	1



1. (date, hour, #Origen)

2. (date, hour, #Receiver)

All 3 attributes are required to identify. If we drop anyone, there are repetitions

# Primary Key

**Primary Key (PK):** CK selected by the database designer  
**An entity without PK is NOT WELL-DEFINED**

- Member

Member code

DNI		Date	Member Code	Name
3676373L		1-1-22	123	Juan
4748474P		1-1-22	333	Juan
6727271Q		1-1-23	667	Pere

#Origin	Cost	Hour	Date	#Receiver	Min
66234560	0.8	10:00	1-9-23	93581444	1
93581444	0.4	10:20	1-9-23	66234501	5
66234501	0.8	10:00	2-9-23	93581444	1

- Phone call

(date, hour, #Origen)

## 2.2 Attributes

2.1.1 Definition

2.1.2 Domain

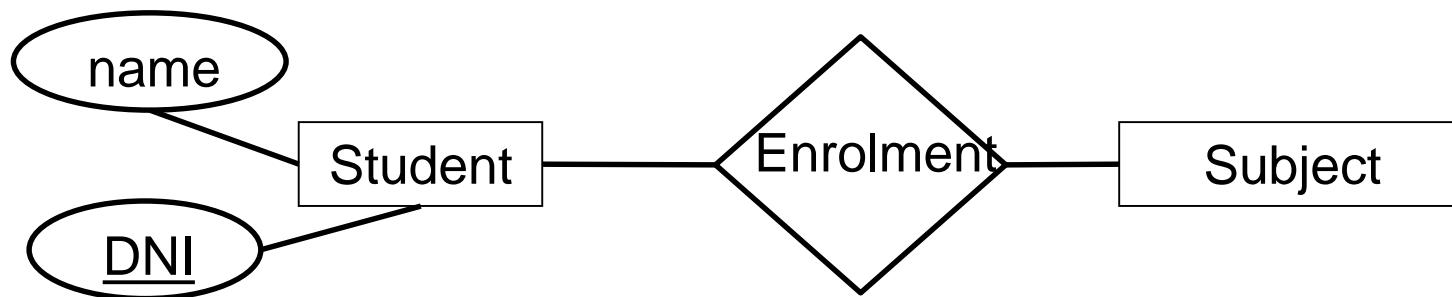
2.1.3 Attributes types

# Definition

Characteristics that describe each entity / relationship. They are represented by ellipses

Their values define the instances of the entities that is part of the information that is stored in the DB.

OBS: The attribute values can change over time



DNI	Name
3676373L	Pere
4748474P	Juan
6727271Q	Pere

Student  
instances

# Domain

Each attribute has associated a domain or values set (type and range) permitted

## Attributes



Name

Date of Birth

Phone number

Penalties

## Domain

Set of strings with length less than 50

Set of strings with the format  
“DD/MM/YYYY”

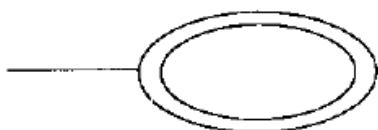
Set of all 9-digit integers

Set of {1, 0}

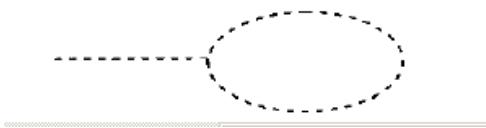
# Attribute types



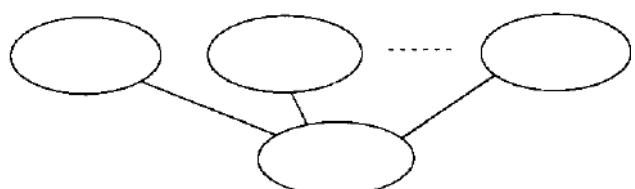
**PK:** set of attributes that uniquely identify entity's instances



**Multivalued:** the attribute has more than one value for each instance (they are "vectors" of variable length for each instance)



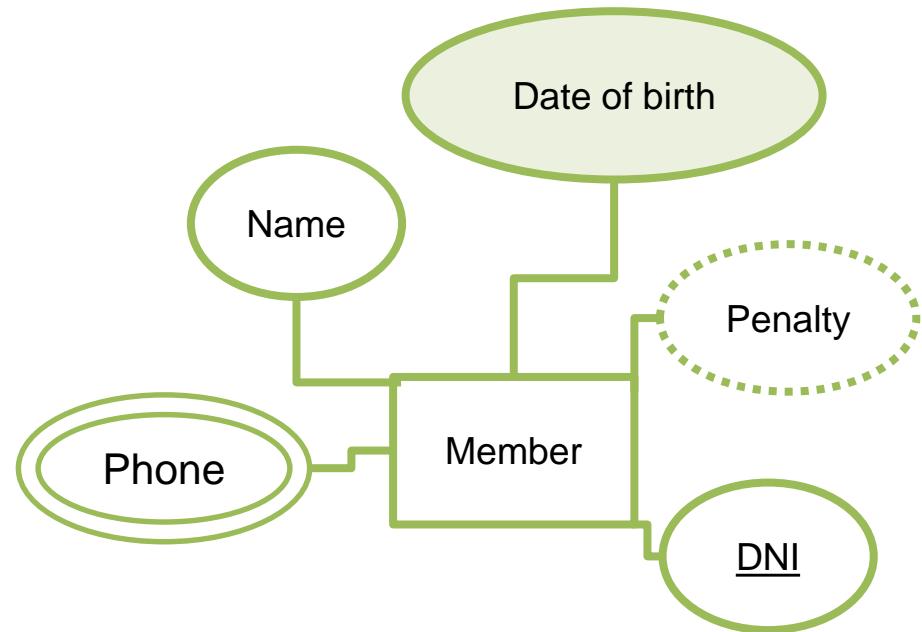
**Derived:** Can be computed from the values of the other attributes



**Composite:** Can be splitted into simpler attributes

# Examples

Multivalued:

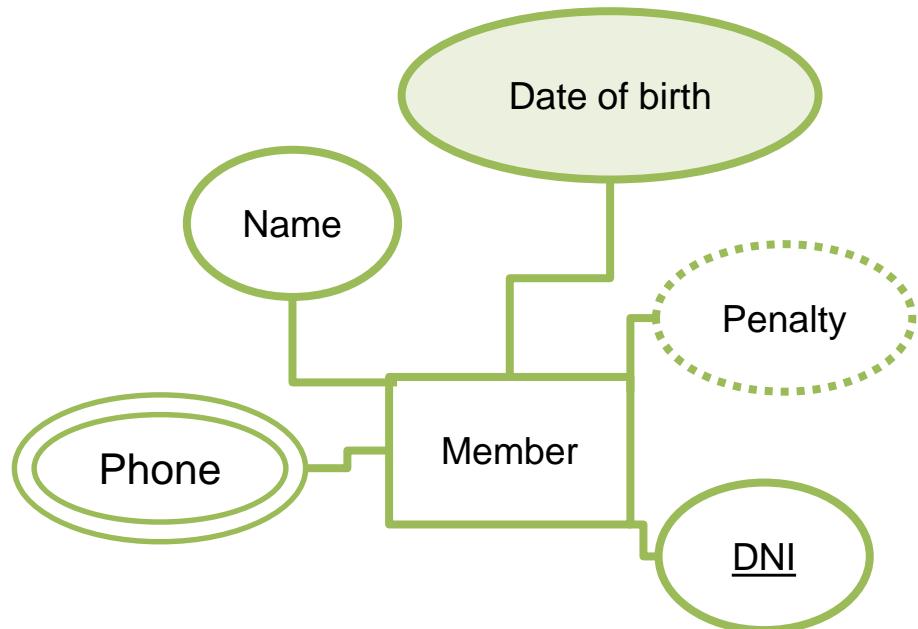


A member can have more than one phone number.  
Dani Alves has 3 phone numbers and Peio has only one

DNI	Name	Surname	Phone
3676373L	Peio	Artola	938373893
4748474P	Dani	Alves	617232066, 623344112, 617734333

# Examples

Composite:

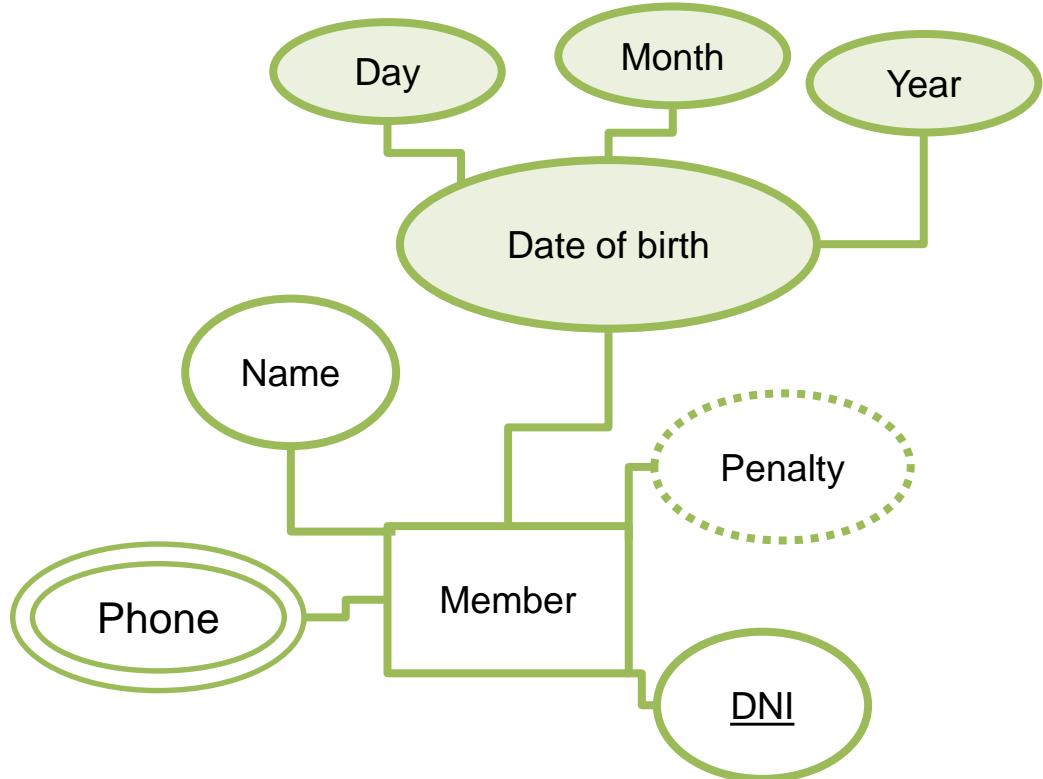


Date of birth can be splitted into  
day, month and year

DNI	Name	Surname	Phone	Date of birth
3676373L	Peio	Artola	938373893	23-Jan-1965
4748474P	Dani	Alves	617232066, 623344112, 617734333	12-Jun-2001

# Examples

Composite:



DNI	Name	Surname	Phone	Birth day	Birth month	Birth year
3676373L	Peio	Artola	938373893	23	1	1965
4748474P	Dani	Alves	617232066, 623344112, 617734333	12	6	2001

# Examples. Composite vs Entity

In the composite attributes the pieces are always of the same type. A composite attribute with different domains, surely should be an entity

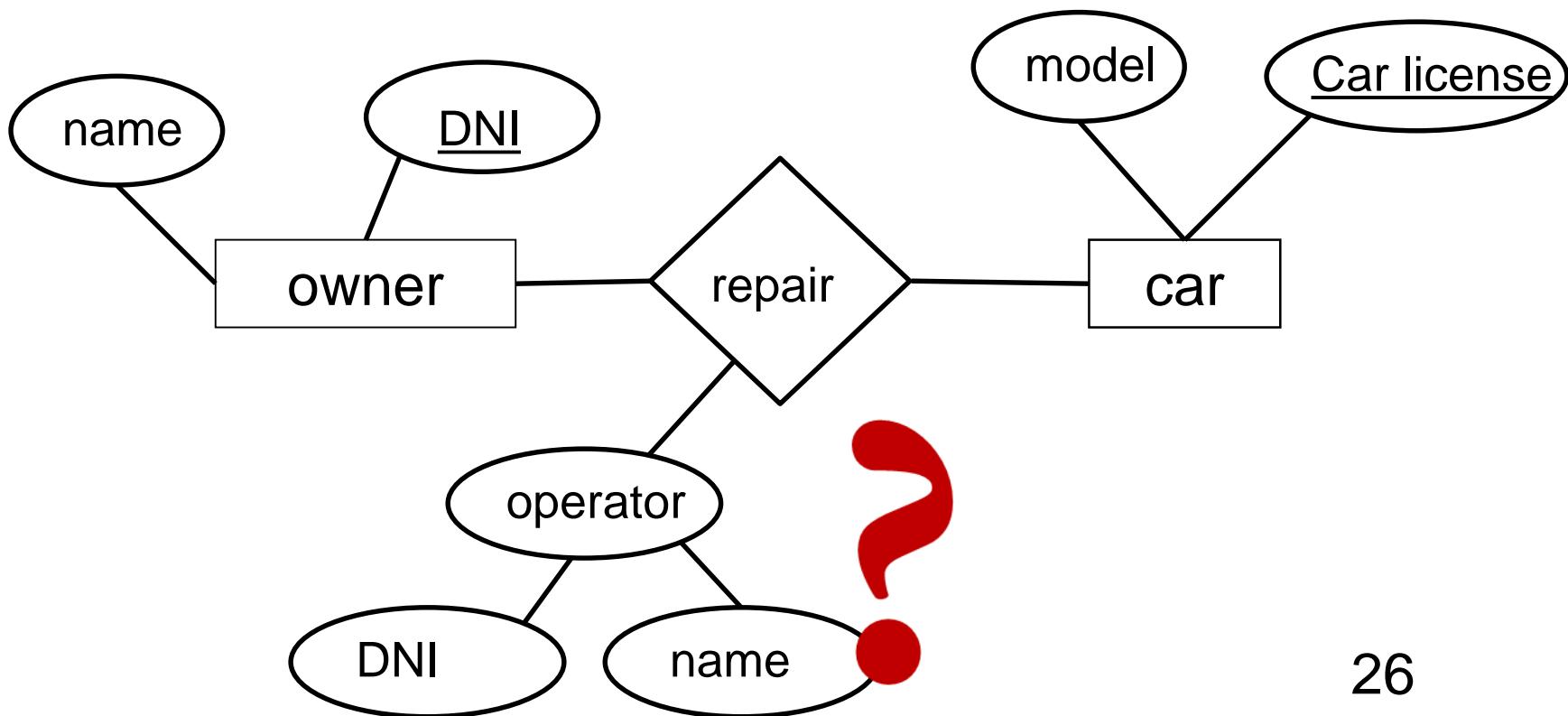
## Car Repair Shop:

A car repair shop wants to keep information about its customers, and repairs. We want to keep the ID and name of the clients, the attendant and the car being repaired. We want to save the name, ID and category of the operator. Of the car, the license plate, model, who repairs it and to whom it belongs.

## Example. Composite vs Entities

### Car Repair Shop:

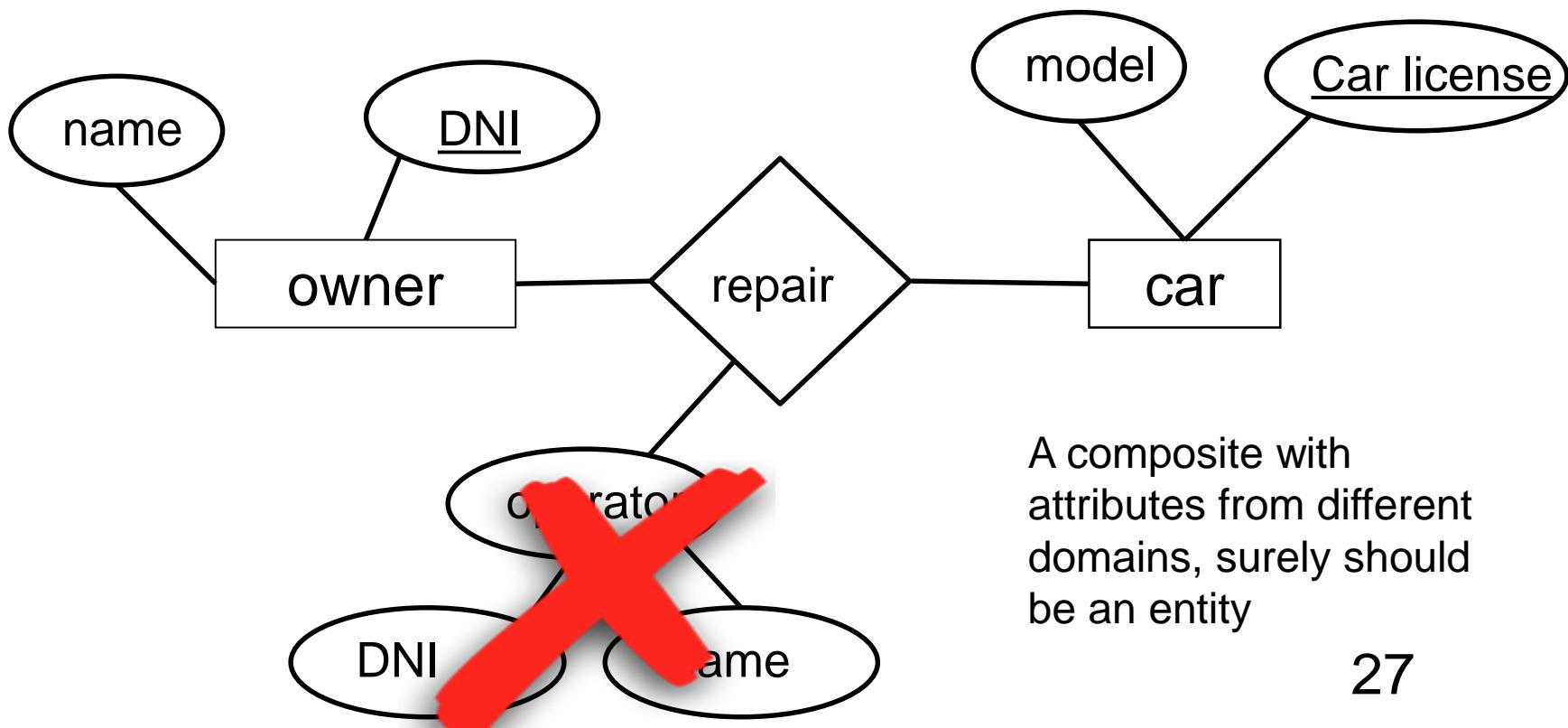
A car repair shop wants to keep information about its customers, and repairs. We want to keep the ID and name of the clients, the attendant and the car being repaired. We want to save the name, ID and category of the operator. Of the car, the license plate, model, who repairs it and to whom it belongs.



## Example. Composite vs Entities

### Car Repair Shop:

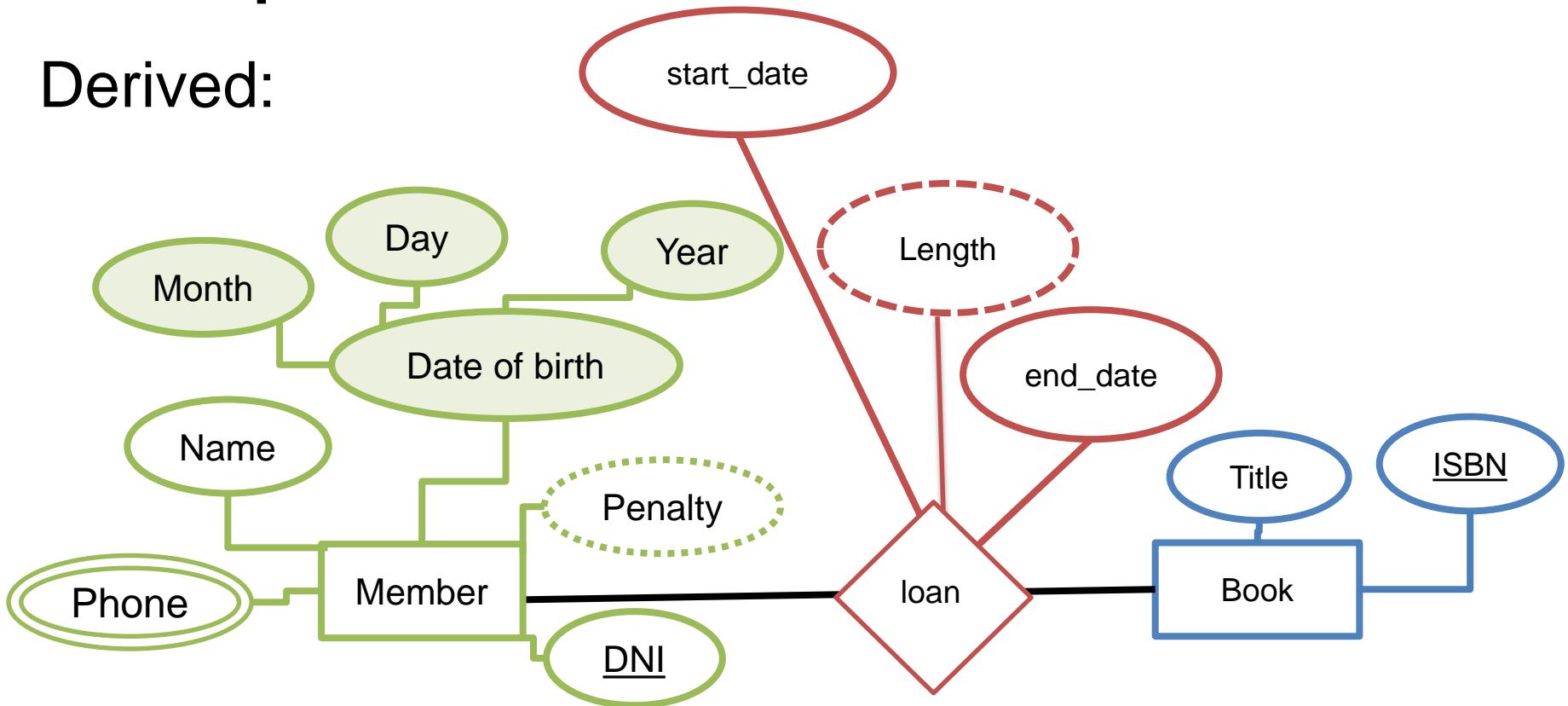
A car repair shop wants to keep information about its customers, and repairs. We want to keep the ID and name of the clients, the attendant and the car being repaired. We want to save the name, ID and category of the operator. Of the car, the license plate, model, who repairs it and to whom it belongs.



A composite with  
attributes from different  
domains, surely should  
be an entity

# Examples

Derived:



The penalty can be obtained from the list of overdue books represented by the relation

Length is the difference between start date and end date

# BLOCK 2

# DESIGN INTRODUCTION

Debora Gil, Oriol Ramos, Carles Sánchez

# Outline

0. Review of previously discussed
1. Design phase
2. Conceptual Design
  - 2.1 Basic Structures
  - 2.2 Properties of the links

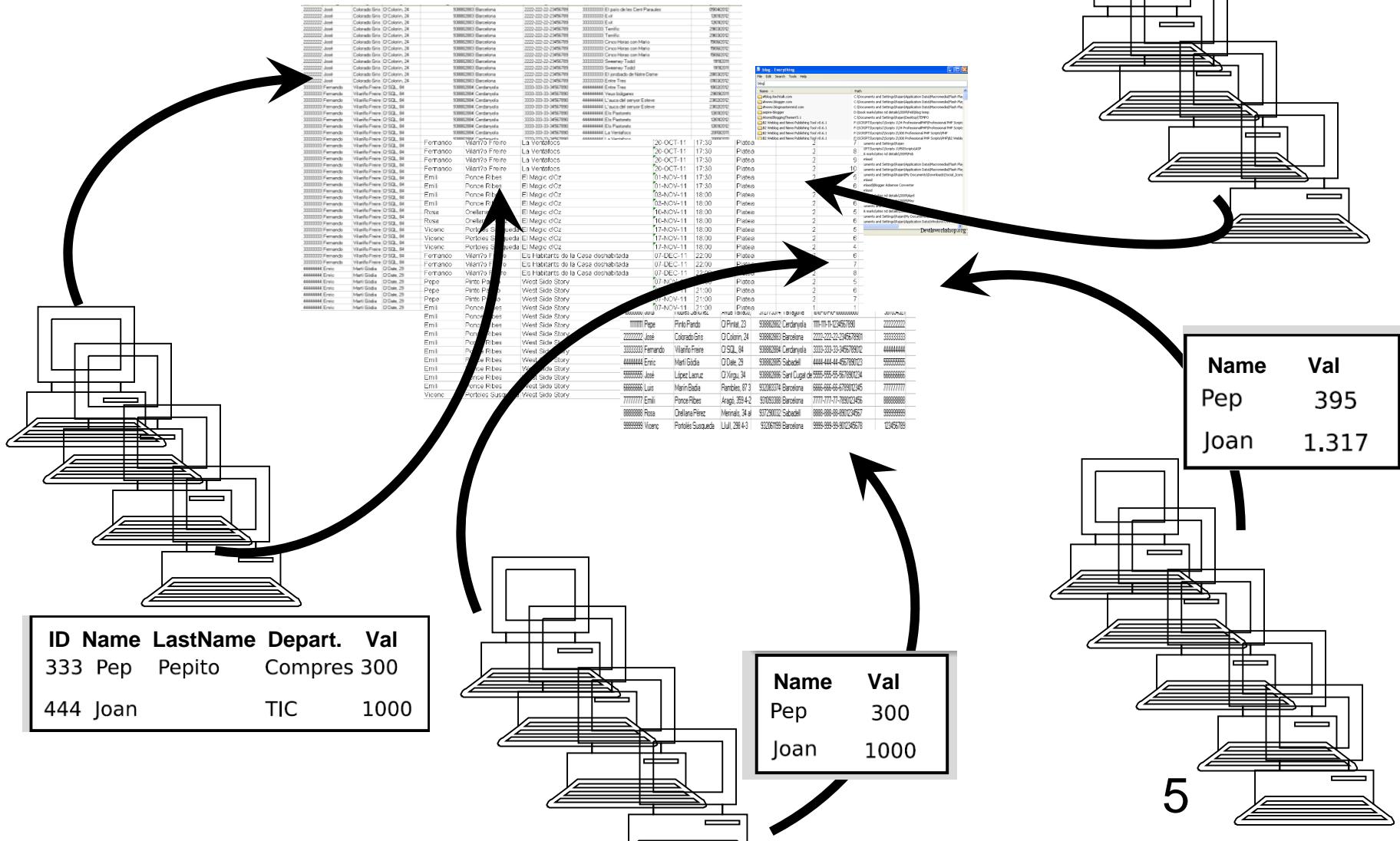
# Outline

**0. Review of previously discussed**

**1. Design phase**

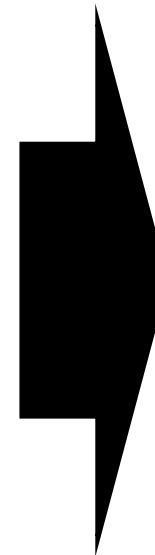
# 0. Review

# Computational storage systems to manipulate volumes of related data in centralized systems (multi-user)

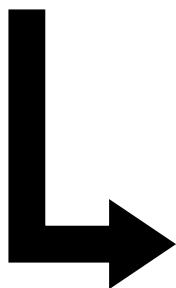


# Key points of a DBS

- **Data integrity** (consistency and coherence of the content)
- **Independency** program/ format data
- **Efficient** system (access to data)



Description at  
Conceptual Level  
(Relational Model  
and E-R Diagram)

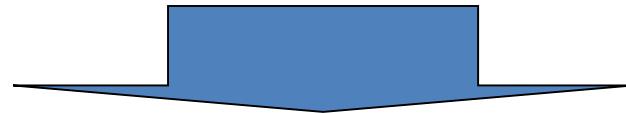


Physical Level description (file  
system and devices, DBM)

# Data description

The DBS must satisfies the necessities of information to different users who share the data

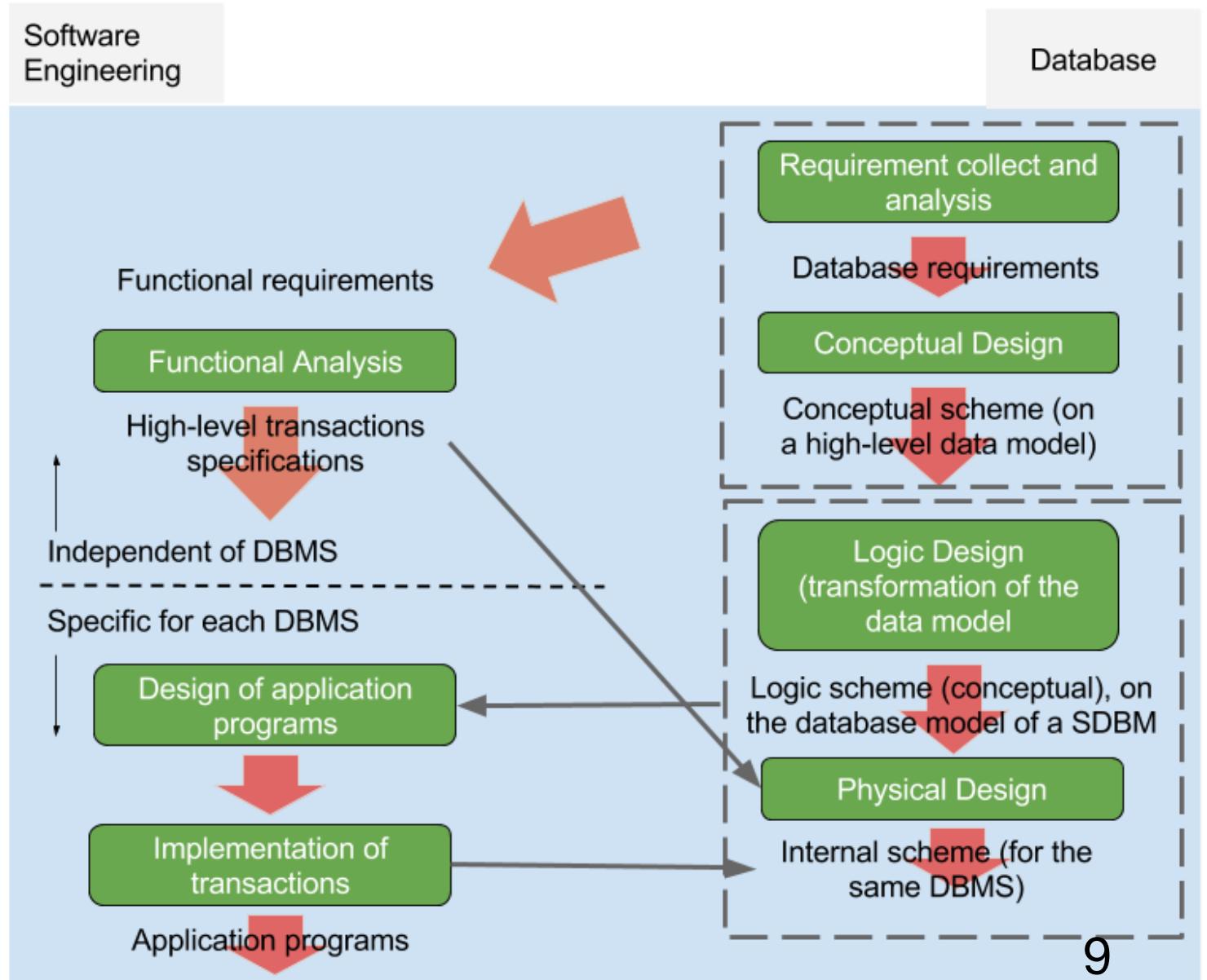
Description of the data (according to user's requirements) guaranteeing the efficiency and consistency



## **Design of the Database**

# 1. Design phases

# Design of an Application



# Design phases of a DB

- Capture and analysis of requirements
  - Conceptual design
- 

- Logic design

- Physical design

Specific for each  
DBMS

# Requirements

To characterize the DB user necessities, they data and they usage:

- Data requirements (what information we have): Description of data and the relationship between them
- Functional requirements (what we want to do with the data): Description of the operations (transactions) to be carried out with the data

# Conceptual Design

Translation of the data requirements to the abstract model that has been chosen (conceptual scheme) :

ER Diagram  
Descriptive report

It is necessary to validate its functionality according the transactions indicated in the functional requirements (set of tests)

Is an abstract description (high-level model) of independent data of the DBMS

# Abstract Data Models

They allow semantically to model the data and links (relationships, inter-relations) that exist between them

Developed to increase the effectiveness and accuracy of the DB design

- Existing models:
- Binary-Semantic (Abrial, 1974)
  - **Entity-Relationship** (Chen, 1976)
  - Semantic Data Model–SDM (Hammer-McLeod, 1981)
  - Functional (Shipman, 1981)
  - Object Oriented-**extended ER model**

# Logical Design

Translation of conceptual design to the data model (generally relational) of the DBMS

Precise implementation depends of the DBMS of our application:

Table diagram, SQL implementation → DBM

Classes Diagram, UML implementation → Functional Analysis (Software Engineering)

Object Oriented DB (ODL, OSL) → Management and administration of a DB

# Physical Design

Structure the tables/classes in files and devices.

Depends of the DBMS and the selected operating system

The device distribution must guarantee the efficiency in access and “space”.

Depends of the transactions specified in the functional requirements.

# Block 2

# BASIC E/R DESIGN

## (PART 2)

Debora Gil, Oriol Ramos, Alejandro Párraga, Carles Sánchez

# Basic ER Design Contents

1. E-R Model Introduction

2. Basic Structures

    2.1 Entities

    2.2 Attributes

    2.3 Relationships

3. Relationships Features

    3.1 Cardinality

    3.2 Degree

    3.3 Participation

# Basic ER Design Contents (Part II)

1. E-R Model Introduction

2. Basic Structures

    2.1 Entities

    2.2 Attributes

    2.3 Relationships

3. Relationships Features

    3.1 Cardinality

    3.2 Degree

    3.3 Participation

## 2. Basic Structures

### 2.3 Relationships

## 2.3 Relationships

# Definition

Association between different related entities instances. They are represented by rhombuses

Each instance of the relationship is defined by the values of the PKs of the associated entities instances

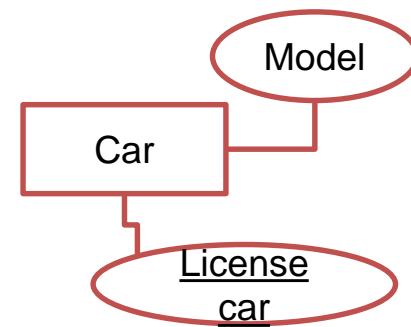
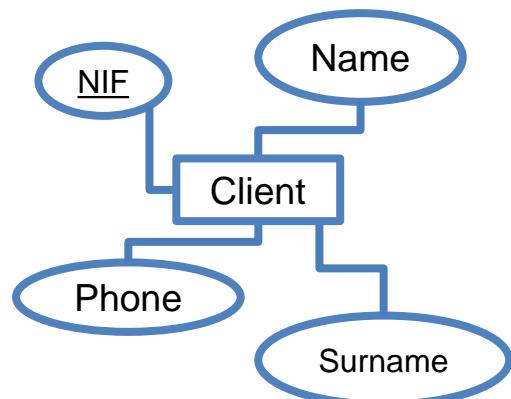
**Relationships NEVER have PKs** (they are already uniquely defined by the PKs of the two related instances)

## Client

NIF	Name	Surname	Phone
3676373L	Peio	Artola	938373893
4748474P	Dani	Alves	617232066
1233399Q	Alex	Morera	617188819
...	...	...	...

## Car

Model	License car
Ka	3090 BKJ
Focus	7839 JKH
Escort	6677 PPL
...	...

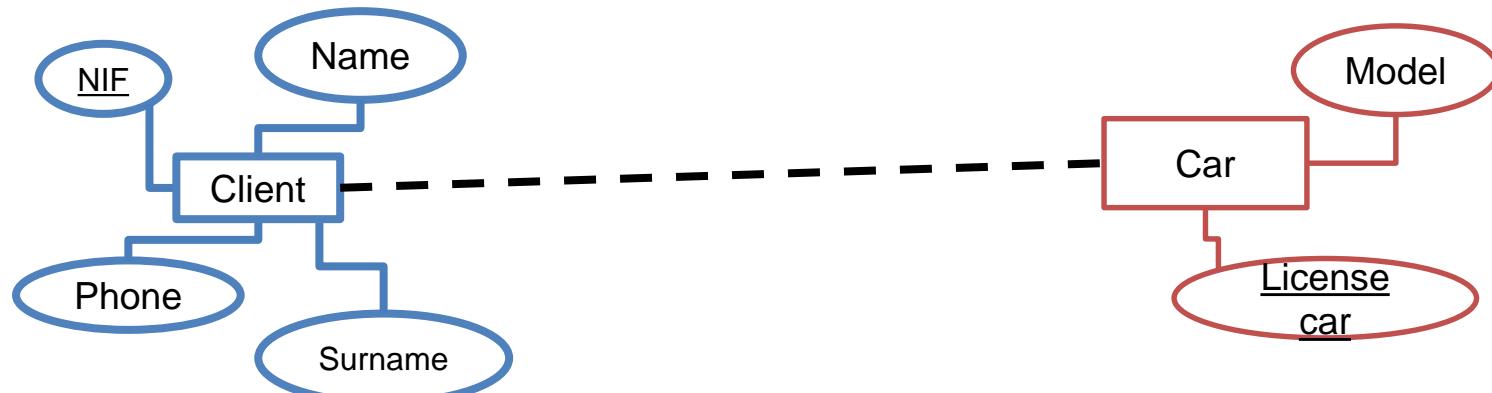


## Client

NIF	Name	Surname	Phone
3676373L	Peio	Artola	938373893
4748474P	Dani	Alves	617232066
1233399Q	Alex	Morera	617188819
...	...	...	...

## Car

Model	License car
Ka	3090 BKJ
Focus	7839 JKH
Escort	6677 PPL
...	...



# Correlation with the real world:

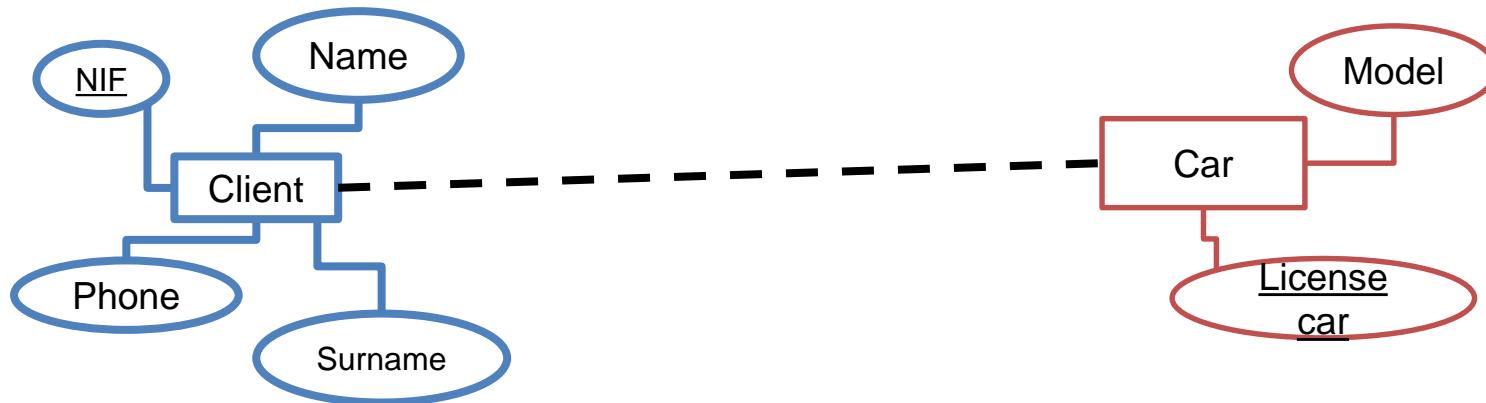
Relationships are correlated with the real world

Client

NIF	Name	Surname	Phone
3676373L	Peio	Artola	938373893
4748474P	Dani	Alves	617232066
1233399Q	Alex	Morera	617188819
...	...	...	...

Car

Model	License car
Ka	3090 BKJ
Focus	7839 JKH
Escort	6677 PPL
...	...



## Real World:

- Peio Artola buys a Ka with license 3090 BKJ
- Dani Alves buys a Focus with license 7839 JKH and an Escort with license 6677 PPL
- ...

# Symbolic representation:

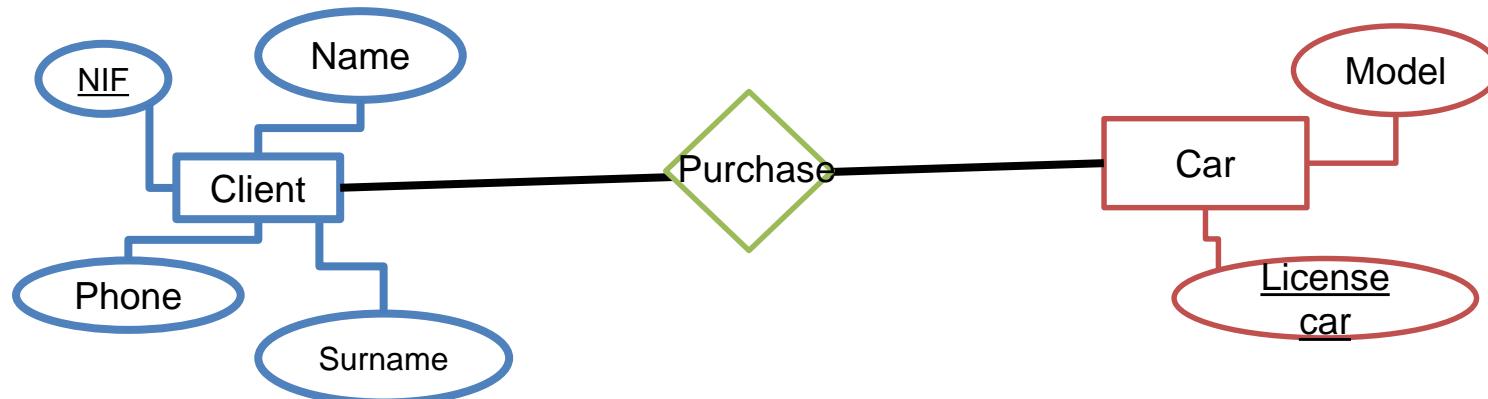
Relationships are represented by rhombuses

Client

NIF	Name	Surname	Phone
3676373L	Peio	Artola	938373893
4748474P	Dani	Alves	617232066
1233399Q	Alex	Morera	617188819
...	...	...	...

Car

Model	License car
Ka	3090 BKJ
Focus	7839 JKH
Escort	6677 PPL
...	...



## Real World:

- Peio Artola buys a Ka with license 3090 BKJ
- Dani Alves buys a Focus with license 7839 JKH and an Escort with license 6677 PPL
- ...

## Relationship Attributes:

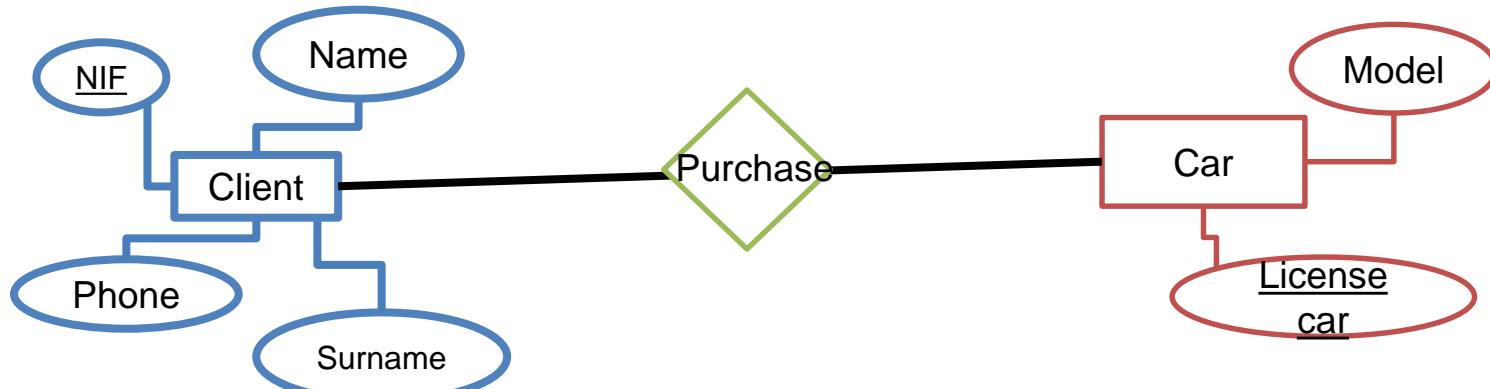
Sometimes it may be appropriate to associate attributes with relationships in themselves ...

Client

NIF	Name	Surname	Phone
3676373L	Peio	Artola	938373893
4748474P	Dani	Alves	617232066
1233399Q	Alex	Morera	617188819
...	...	...	...

Car

Model	License car
Ka	3090 BKJ
Focus	7839 JKH
Escort	6677 PPL
...	...



## Relationship Attributes:

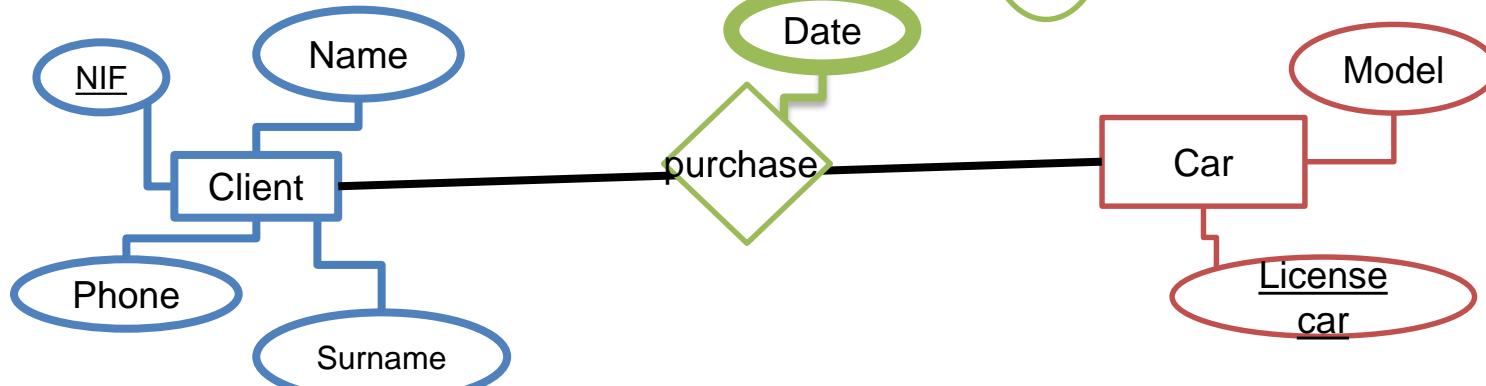
Sometimes it may be appropriate to associate attributes with relationships in themselves ...

Client

NIF	Name	Surname	Phone
3676373L	Peio	Artola	938373893
4748474P	Dani	Alves	617232066
1233399Q	Alex	Morera	617188819
...	...	...	...

Car

Model	License car
Ka	3090 BKJ
Focus	7839 JKH
Escort	6677 PPL
...	...



## Real World:

- Peio Artola buys a Ka with license 3090 BKJ **on 25/01/2015**
- Dani Alves buys a Focus with license 7839 JKH **on 18/06/2016**
- and an Escort with license 6677 PPL **on 24/09/2016**
- ...

# Relationship Primary Key (PK)

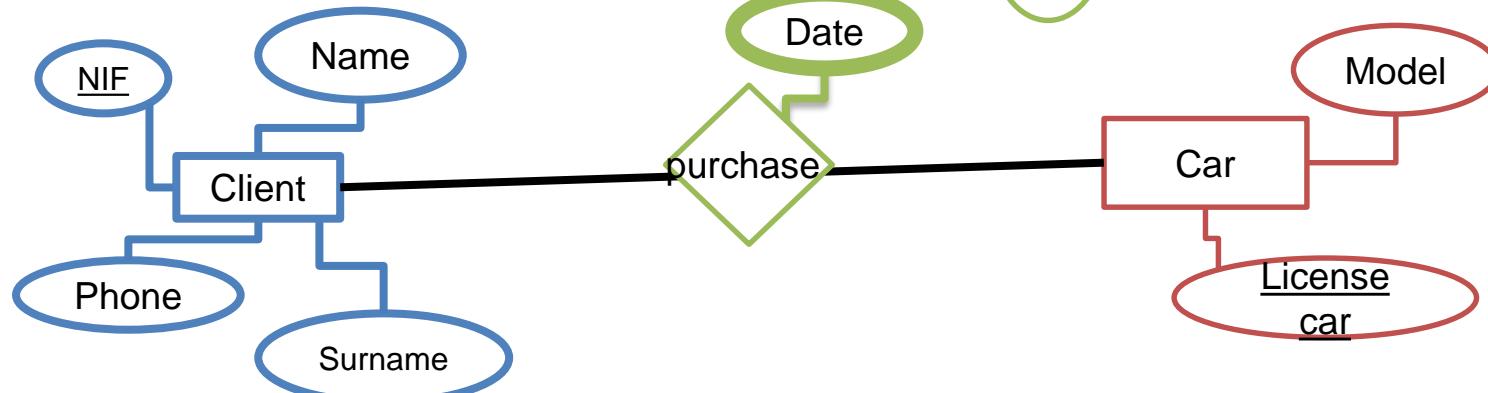
Relationships NEVER have PKs (they are already uniquely defined by the PKs of the two related instances)

Client

NIF	Name	Surname	Phone
3676373L	Peio	Artola	938373893
4748474P	Dani	Alves	617232066
1233399Q	Alex	Morera	617188819
...	...	...	...

Car

Model	License car
Ka	3090 BKJ
Focus	7839 JKH
Escort	6677 PPL
...	...



## Real World:

- Peio Artola buys a Ka with license 3090 BKJ on 25/01/2015
- Dani Alves buys a Focus with license 7839 JKH on 18/06/2016
- and an Escort with license 6677 PPL on 24/09/2016
- ...

# Examples

# Example 1.

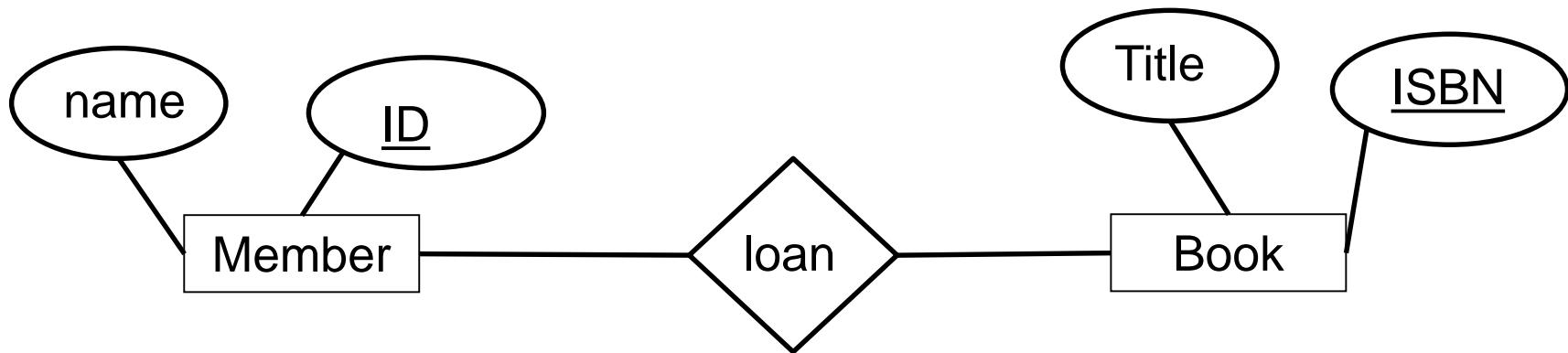
## Information contained in the relationship

### Library

We want to manage a library network loans and know the free copies.  
From a book we want to save the ISBN and the title.  
From a member we want to save name and ID.  
We want to know what books each member has.

Entities??  
Relationships??  
Attributes??

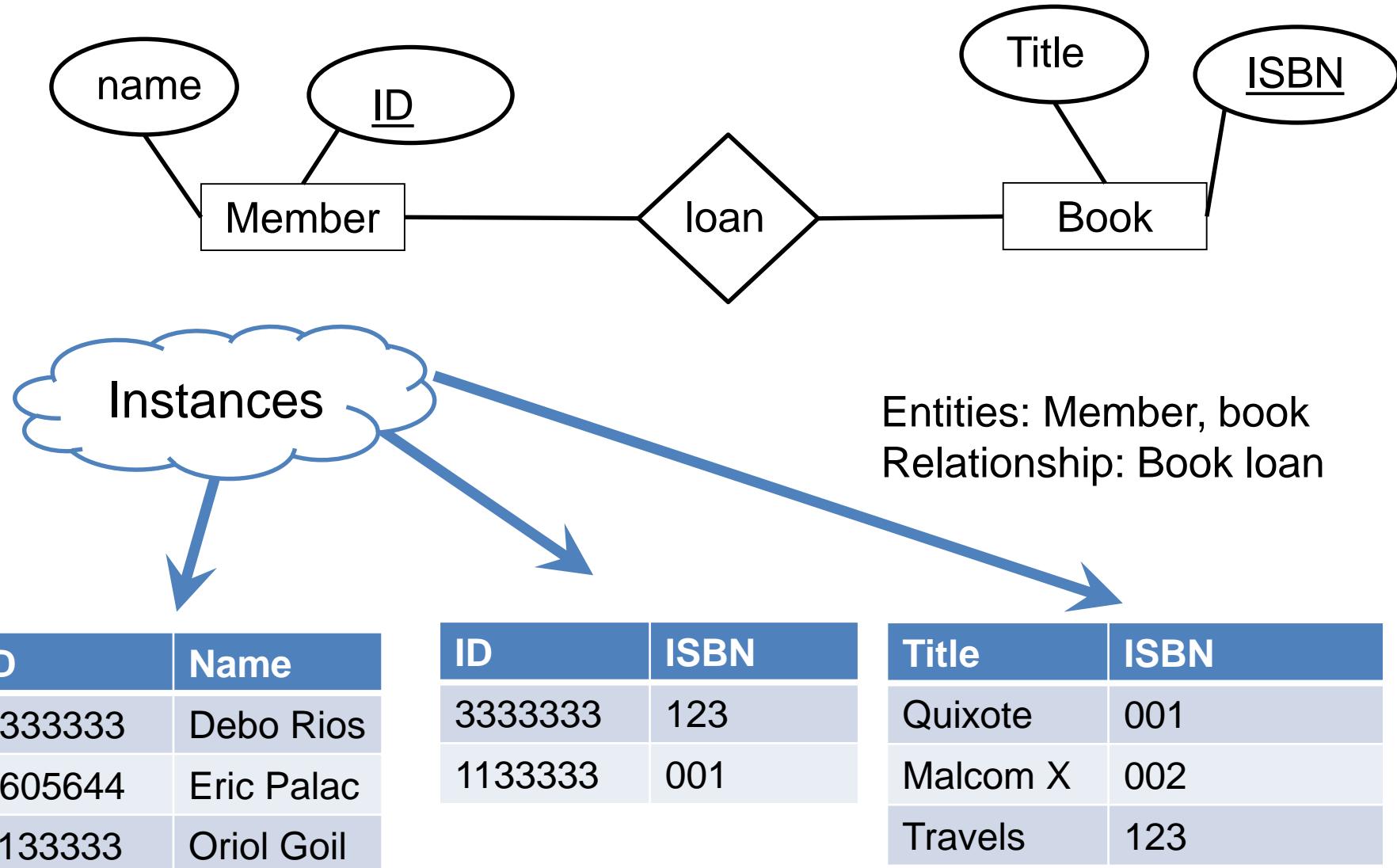
## Example 1. Information contained in the relationship



Entities: Member, book

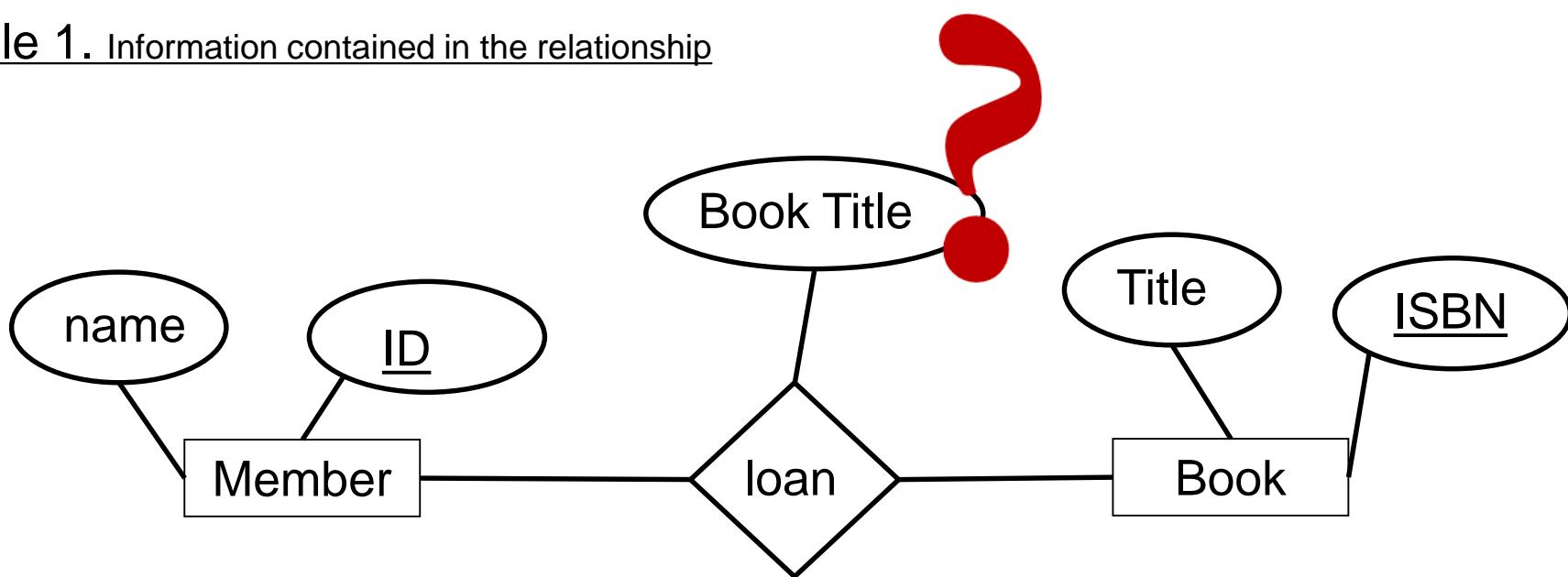
Relationship: Book loan

## Example 1. Information contained in the relationship



Obs: Not all instances must participate in the relationship, only the loaned books.

## Example 1. Information contained in the relationship

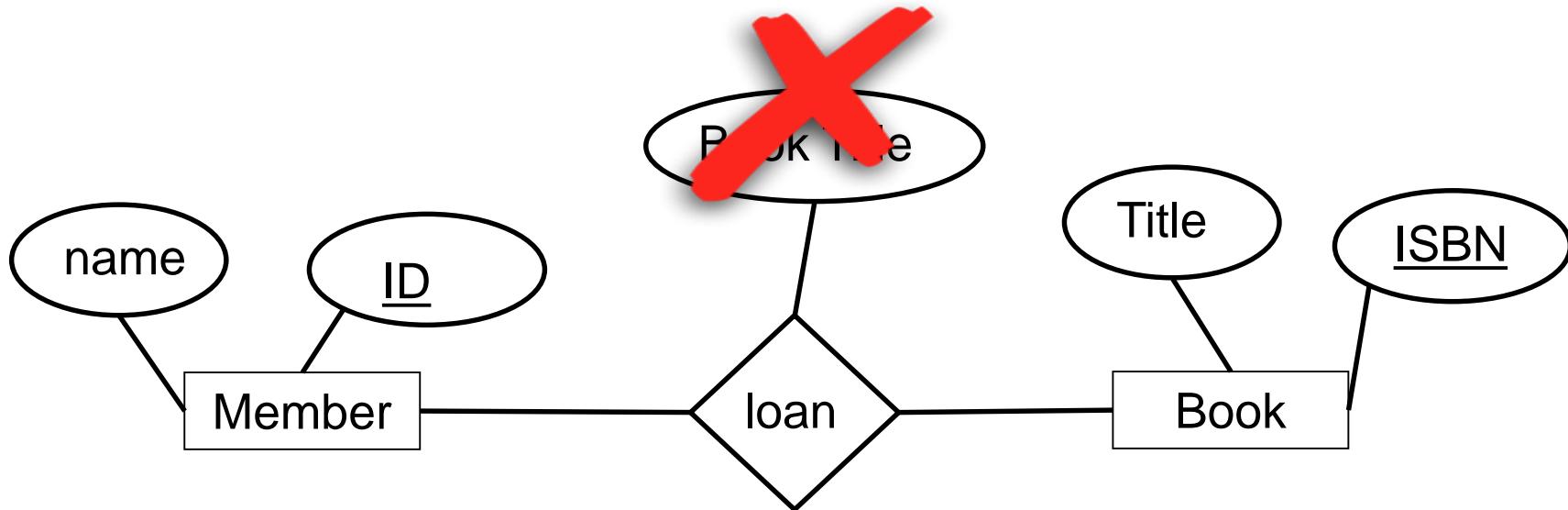


With this design, can we know the titles of the books that are on loan?

How would you modify it?

Should the title of the book be incorporated as an attribute of the "loan" relationship?

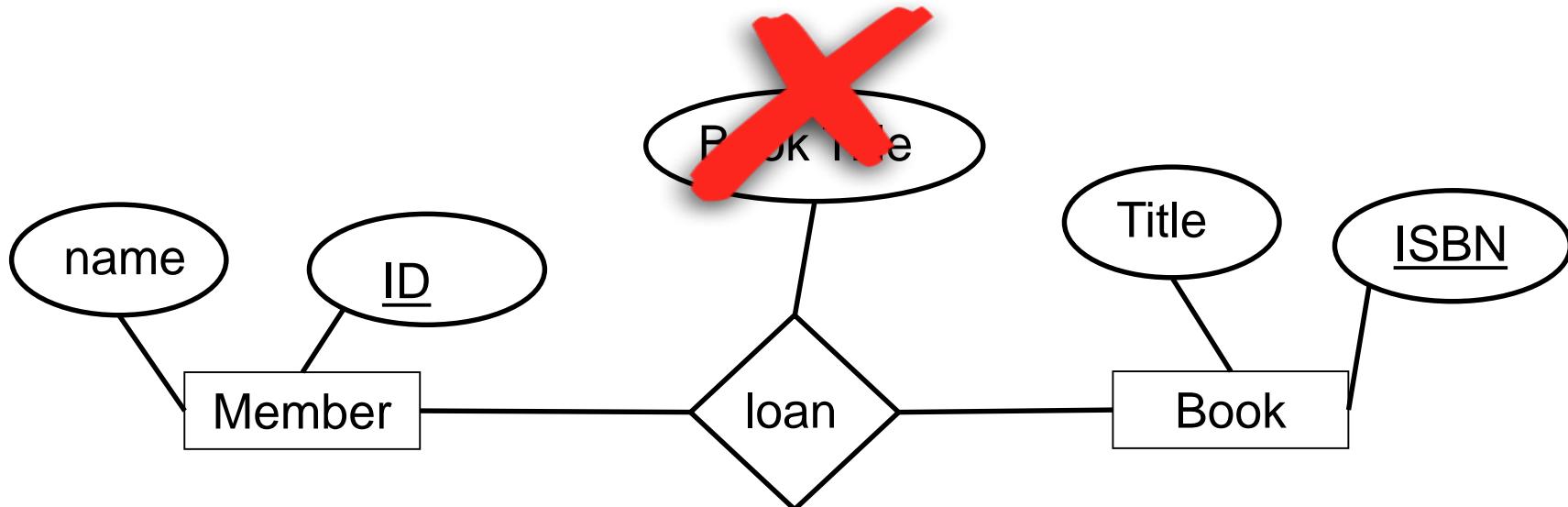
## Example 1. Information contained in the relationship



Relationship contains (indirectly, through the PKs of the related entities) all the information (attributes) of the entities that it relates.

You do **NOT** need attributes to the link that are already in the related entities.

## Example 1. Information contained in the relationship



In addition, these repetitions introduce redundancy of attributes that the DBMS will not handle (updates, FK modifications only) and may violate integrity

Member
name
<u>ID</u>

$ID_{loan} = ID_{Member}$   
guaranteed by integrity that is  
managed by DBMS

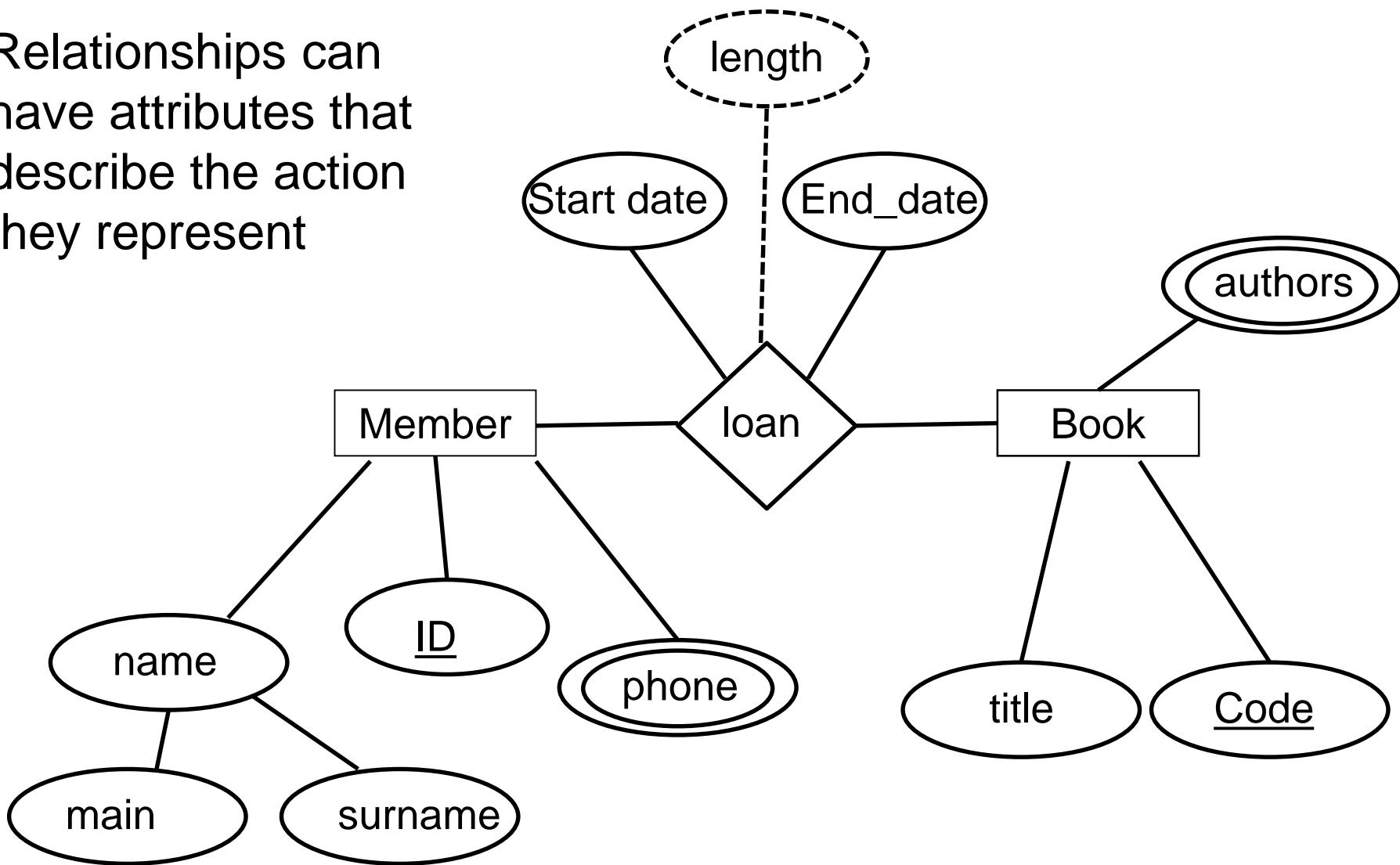
Loan
Book_Title
<u>ID</u>
<u>ISBN</u>

Book
Title
<u>ISBN</u>

$ISBN_{loan} = ISBN_{Book}$   
guaranteed by integrity that is  
managed by DBMS

## Example 1. Information contained in the relationship

Relationships can have attributes that describe the action they represent



# Example 3. Identify entities and relationships

## **Database subjects**

The students of this subject are distributed in different groups according to type of teaching (theory, problems and practices).

Each of these courses is taught by a teacher (which can always be the same). In addition, depending on the number of students more than one teacher can do the same type of teaching.

And several evaluation tests will be performed for each type of teaching

Entities?

Relationships?

# Example 3. Identify entities and relationships

## Database subjects

The **students** of this subject are distributed in different **groups** according to type of **teaching** (theory, problems and practices).

Each of these **teachings** is taught by a **professor** (which can always be the same). In addition, depending on the number of students more than one **professor** can do the same type of **teaching**.

And several **evaluation tests** will be performed for each type of **teaching**

## Entities

- Students
- Groups
- Teaching
- Professor
- Evaluation test

## Relationships?

# Example 3. Identify entities and relationships

## Database subjects

The **students** of this subject are distributed in different groups according to type of teaching (theory, problems and practices).

Each of these **teachings** is taught by a professor (which can always be the same).

In addition, depending on the number of students more than one **professor** can do the same type of teaching.

And several **evaluation tests** will be performed for each type of **teaching**

### Entities

- Students
- Groups
- Teaching
- Professor
- Evaluation test

### Relationships

- Students are distributed in groups
- Groups vary by teaching
- Teachers teach various types of teaching
- ...

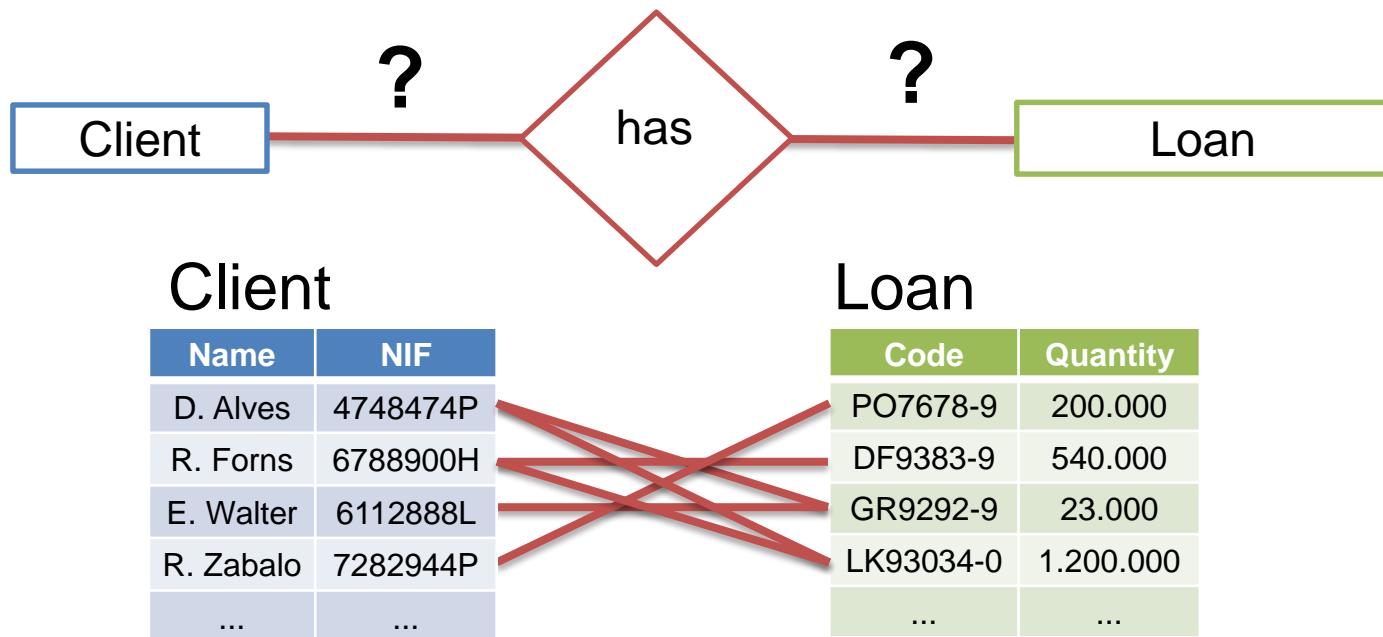
# 3. Relationships Properties

## 3.1 Cardinality

# 3.1 Cardinality

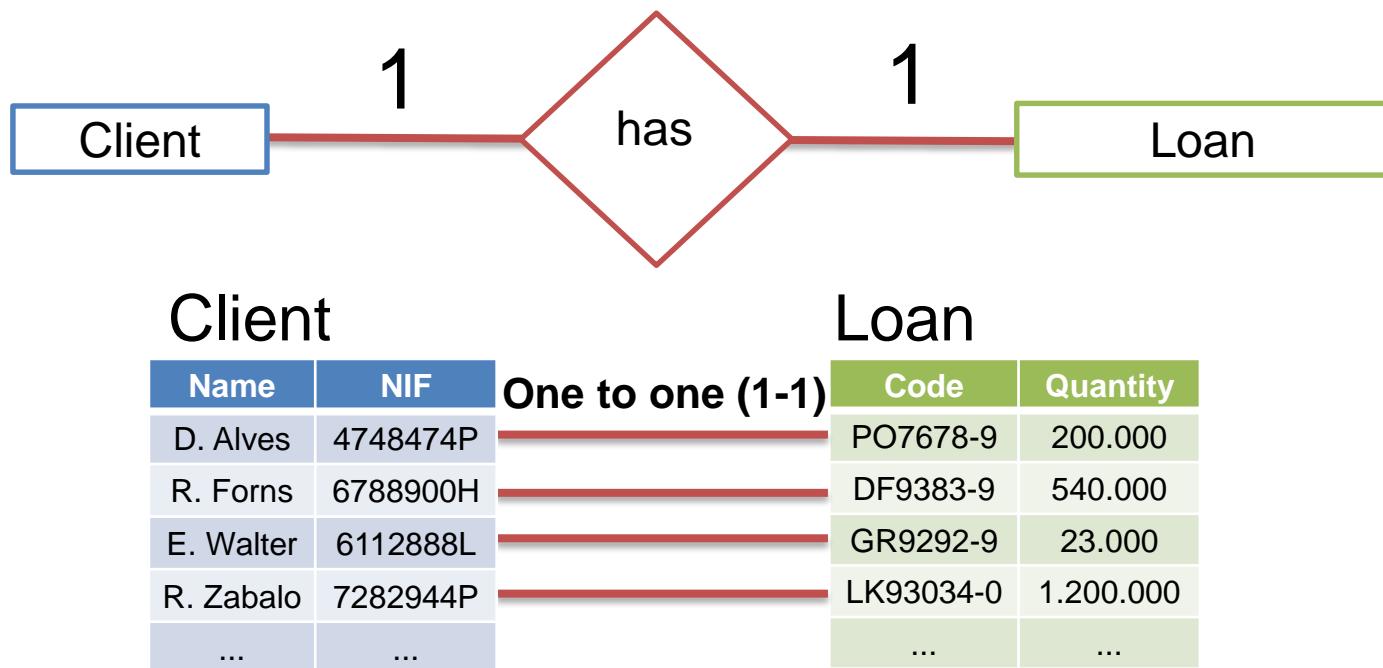
# Definition

**Cardinality** is the maximum number of instances of an entity that may be associated with an instance of the other entity involved in a relationship (there are several types)



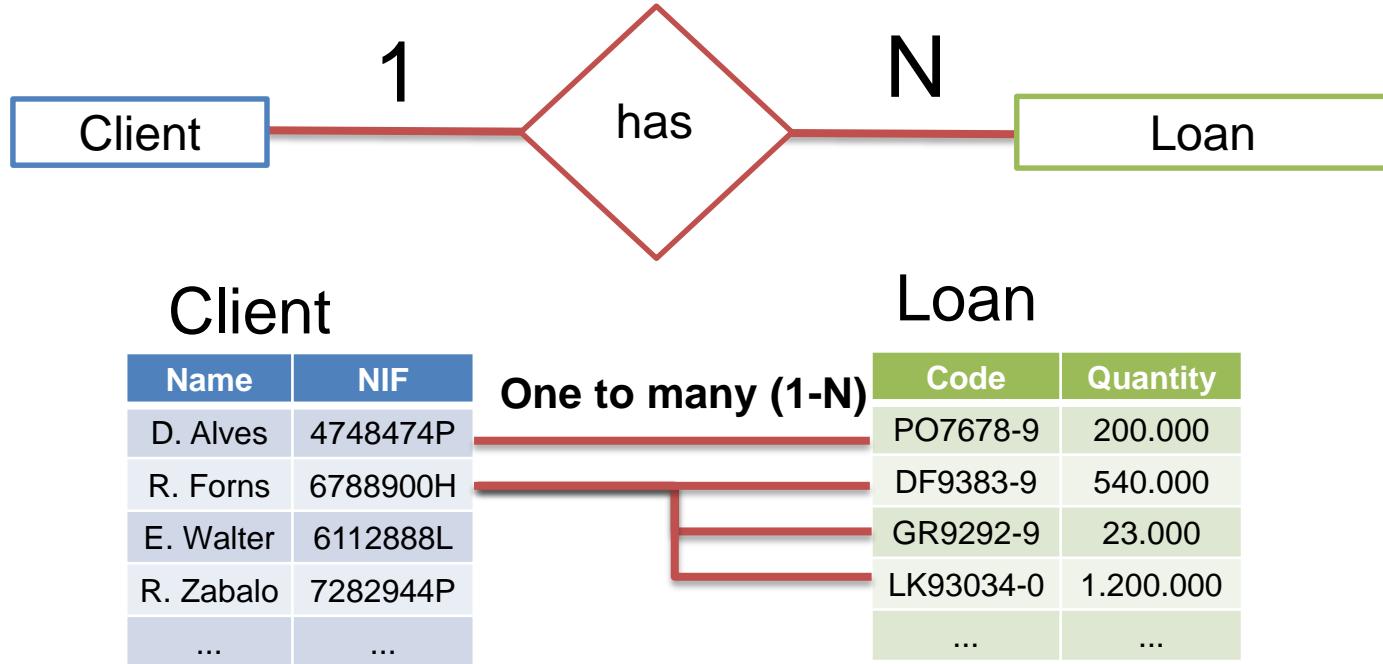
Cardinality is the maximum number of instances of an entity that may be associated with an instance of the other entity involved in a relationship (there are several types)

**Cardinality 1-1:** a client can only have one loan (and one loan can only be took by one client)



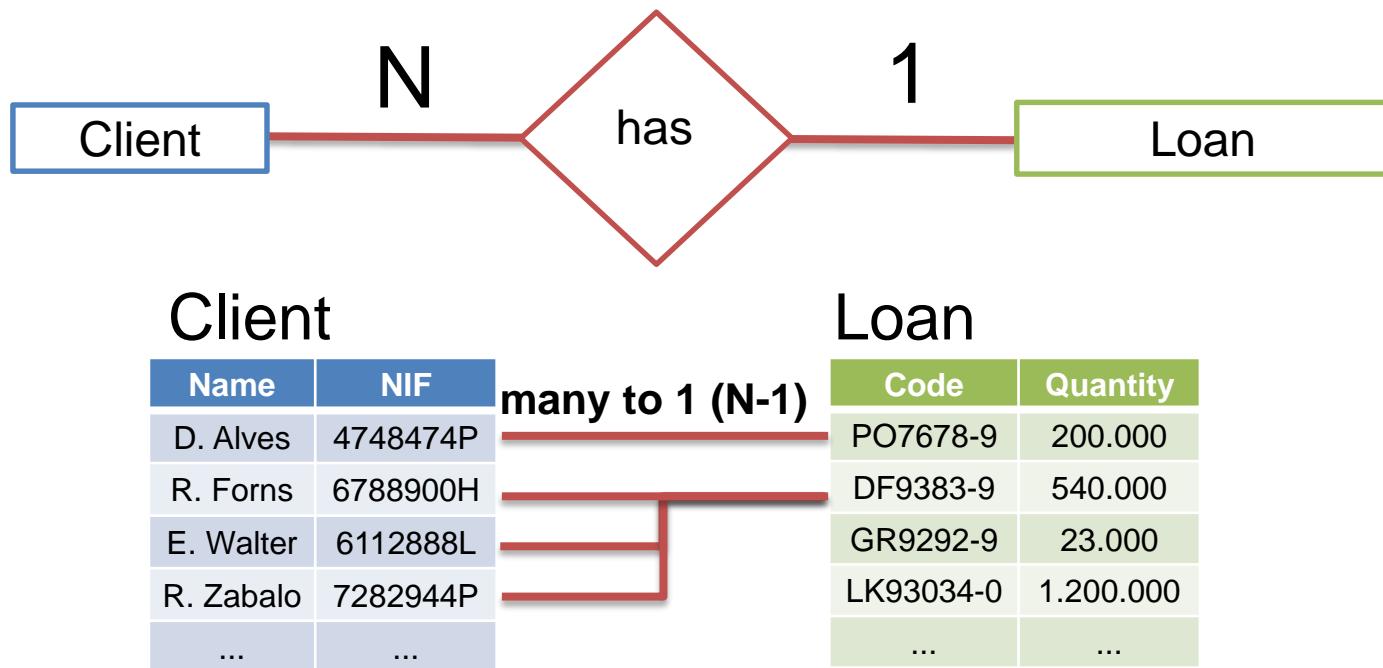
Cardinality is the maximum number of instances of an entity that may be associated with an instance of the other entity involved in a relationship (there are several types)

**Cardinality 1-N:** A client can have more than one loan (and a loan can only be took by a client)



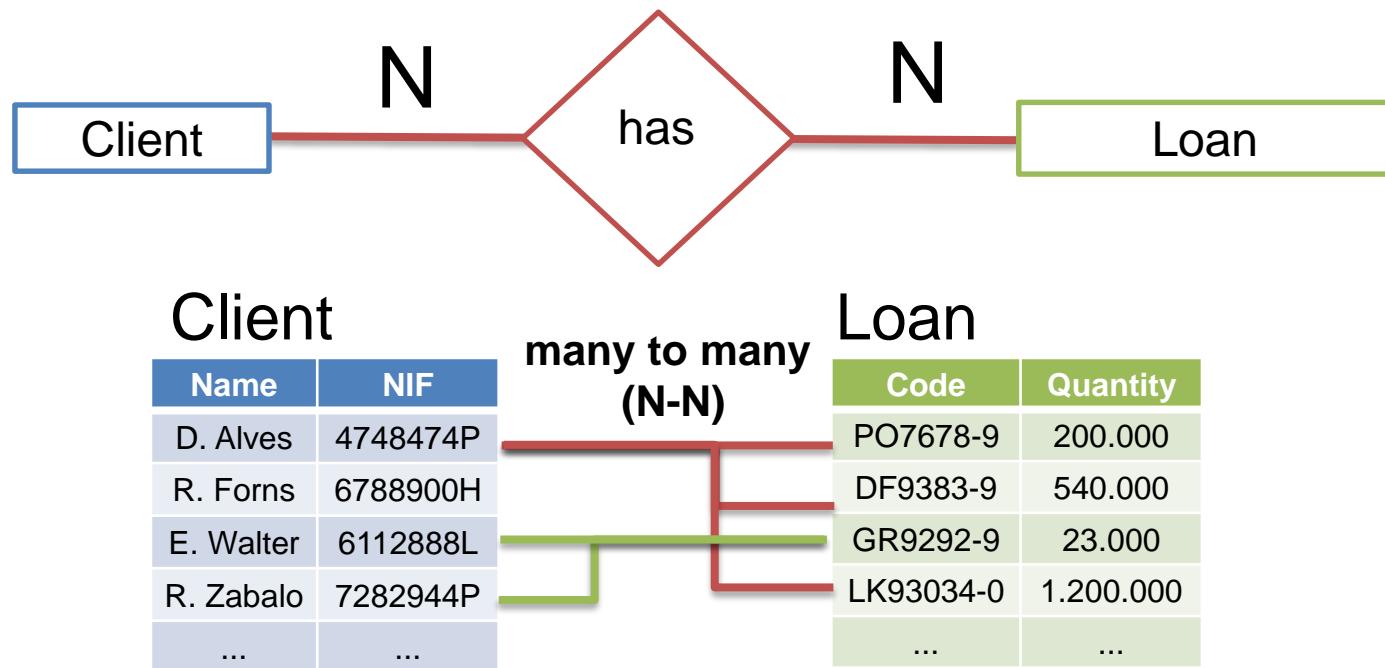
Cardinality is the maximum number of instances of an entity that may be associated with an instance of the other entity involved in a relationship (there are several types)

**Cardinality N-1:** a loan can be shared among multiple clients (but one client cannot have more than one loan)



Cardinality is the maximum number of instances of an entity that may be associated with an instance of the other entity involved in a relationship (there are several types)

**Cardinality N-N:** no restrictions (one loan can be shared between multiple clients and one client can take more than one loan)



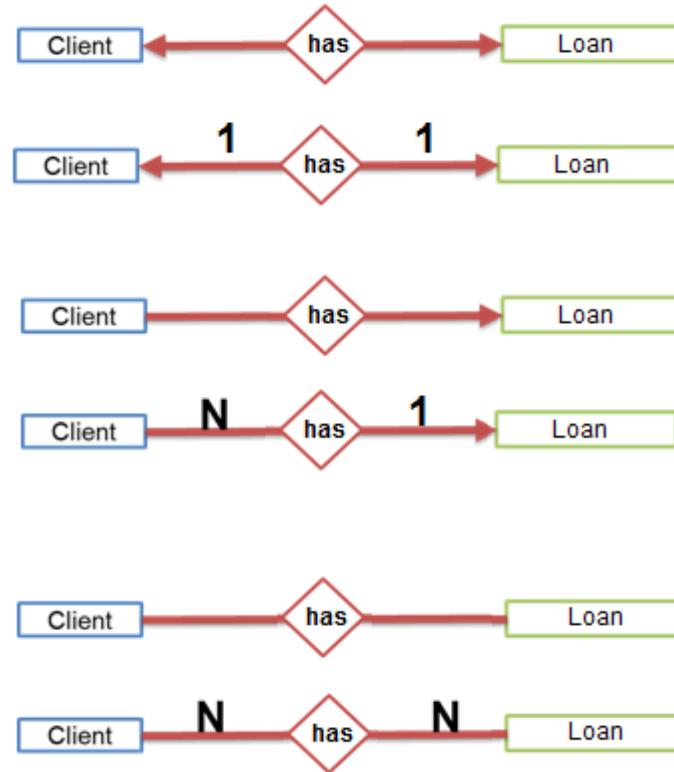
# Symbolic representation:

The simple or directed line (arrow) serves to distinguish between many-to-many, many-to-one, or one-to-many relationships.

The double arrow indicates a **one-to-one** relationship

A simple arrow indicates a **many-to-one** relationship.

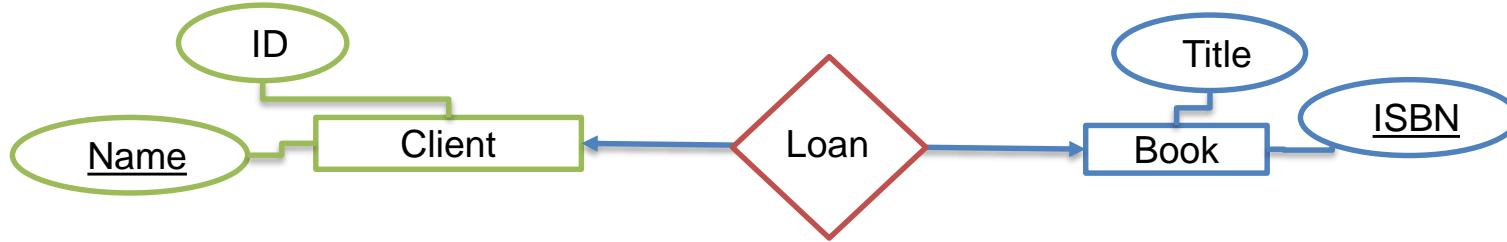
A simple line indicates a **many-to-many** relationship



# Examples

# Example 4. Cardinality type

Consider the E-R design where the "Loan" relationship has **1-1** cardinality



Client

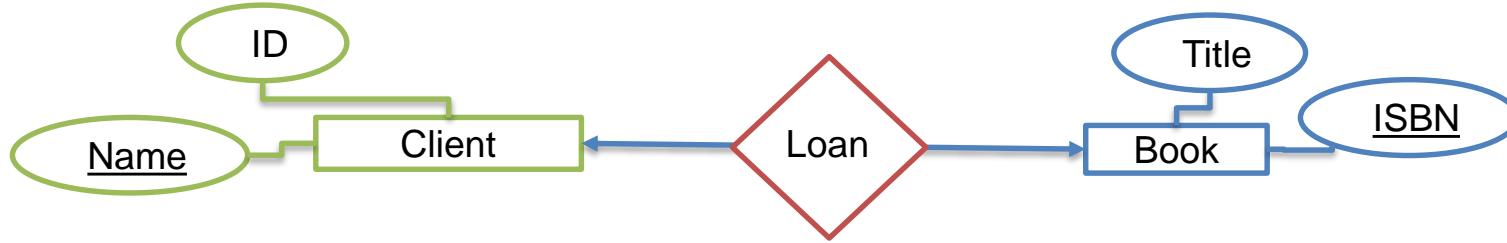
ID	Name
2321323N	Pepe
7484849P	Paco
2342312Q	Kike
4848994J	Montse
...	...

Book

Title	ISBN
Quixot	0011
BBDD	0022
Postres	1122
Hamlet	2829
...	...

# Example 4. Cardinality type

Consider the E-R design where the "Loan" relationship has **1-1** cardinality



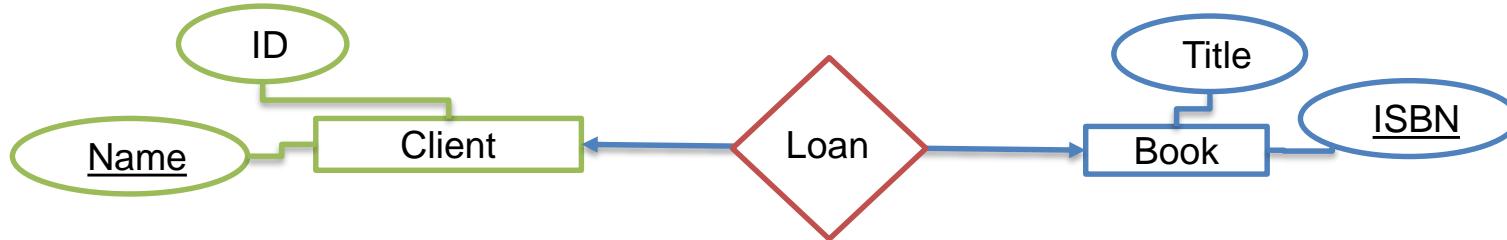
ID	Name
2321323N	Pepe
7484849P	Paco
2342312Q	Kike
4848994J	Montse
...	...

Title	ISBN
Quixot	0011
BBDD	0022
Postres	1122
Hamlet	2829
...	...

#### Example 4. Cardinality type

Consider the E-R design where the "Loan" relationship has 1-1 cardinality

Does this design allow Montse or Paco to take Don Quixote?



ID	Name
2321323N	Pepe
7484849P	Paco
2342312Q	Kike
4848994J	Montse
...	...

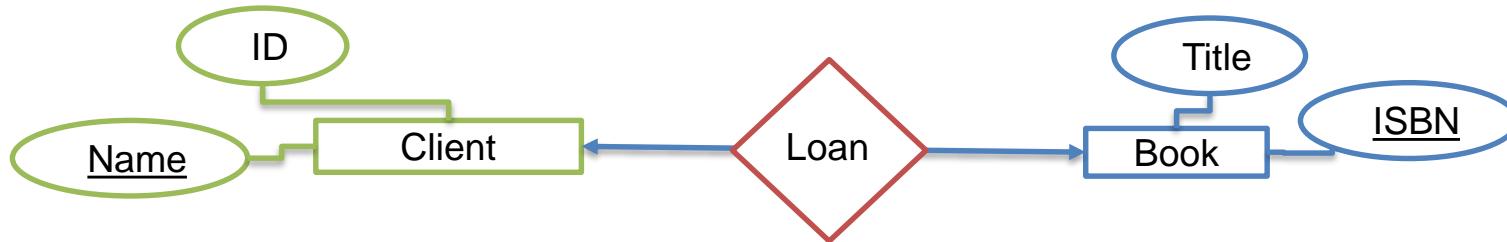


Title	ISBN
Quixot	0011
BBDD	0022
Postres	1122
Hamlet	2829
...	...

#### Example 4. Cardinality type

Consider the E-R design where the "Loan" relationship has 1-1 cardinality

Does this design allow Montse or Paco to take Don Quixote?  
NO: because one book could be loaned by one client.



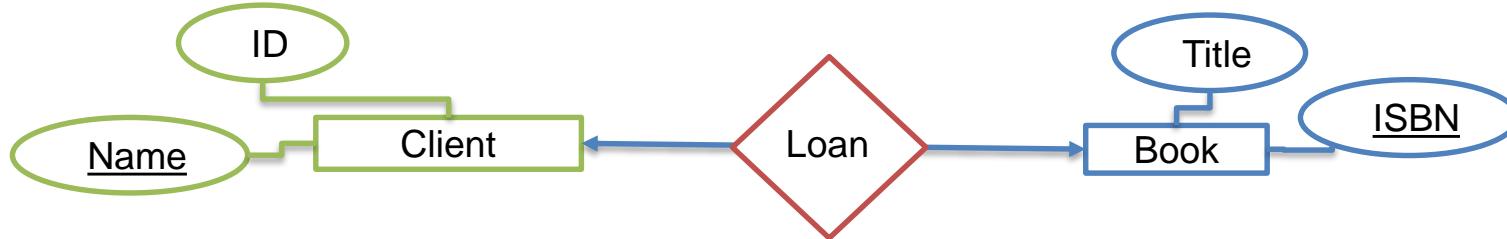
ID	Name
2321323N	Pepe
7484849P	Paco
2342312Q	Kike
4848994J	Montse
...	...

Title	ISBN
Quixot	0011
BBDD	0022
Postres	1122
Hamlet	2829
...	...

#### Example 4. Cardinality type

Consider the E-R design where the "Loan" relationship has **1-N** cardinality

Does this design allow keeping Montse to take Don Quixote?



ID	Name
2321323N	Pepe
7484849P	Paco
2342312Q	Kike
4848994J	Montse
...	...

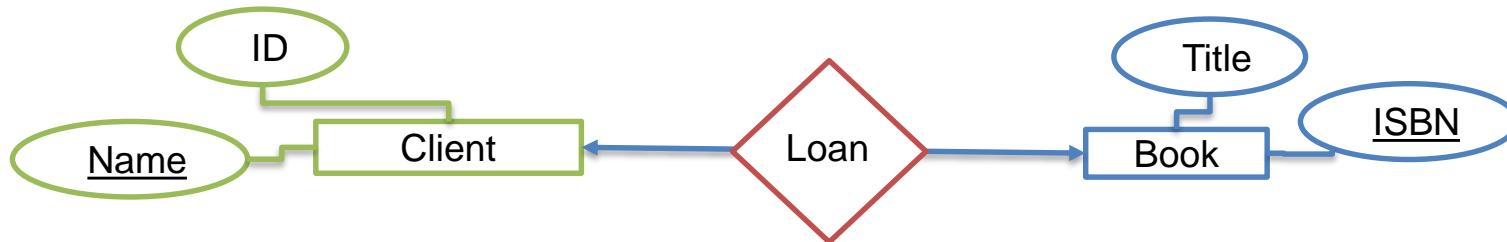


Title	ISBN
Quixot	0011
BBDD	0022
Postres	1122
Hamlet	2829
...	...

#### Example 4. Cardinality type

Consider the E-R design where the "Loan" relationship has **1-N** cardinality

Does this design allow keeping Montse to take Don Quixote?  
YES: One member can have more than 1 book on loan and one book can only be loaned to 1 member



Below the ER diagram are two tables: Client and Book. The Client table has columns ID and Name, with rows for Pepe, Paco, Kike, Montse, and an ellipsis. The Book table has columns Title and ISBN, with rows for Quixot, BBDD, Postres, Hamlet, and ellipses. A large green checkmark is positioned above the Client table. Red dashed lines connect the Client row for Montse to the Book rows for Quixot, BBDD, Postres, and Hamlet, illustrating that Montse can have multiple books on loan.

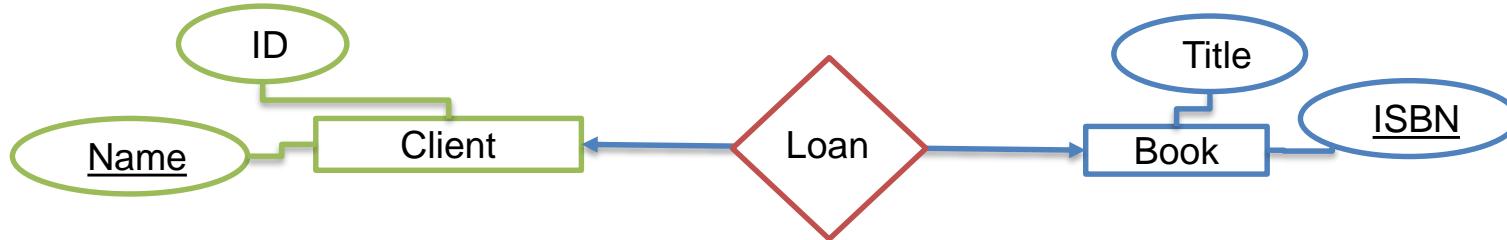
ID	Name
2321323N	Pepe
7484849P	Paco
2342312Q	Kike
4848994J	Montse
...	...

Title	ISBN
Quixot	0011
BBDD	0022
Postres	1122
Hamlet	2829
...	...

#### Example 4. Cardinality type

Consider the E-R design where the "Loan" relationship has **N-1** cardinality

Does this design allow keeping Montse to take Don Quixote?



ID	Name
2321323N	Pepe
7484849P	Paco
2342312Q	Kike
4848994J	Montse
...	...

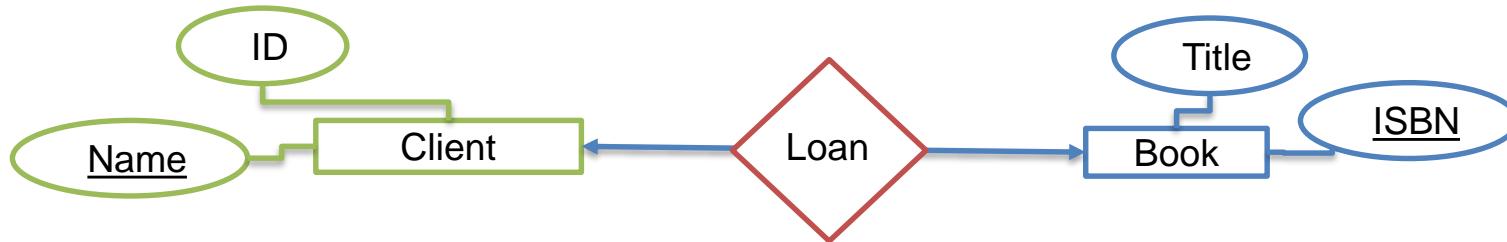


Title	ISBN
Quixot	0011
BBDD	0022
Postres	1122
Hamlet	2829
...	...

#### Example 4. Cardinality type

Consider the E-R design where the "Loan" relationship has **N-1** cardinality

Does this design allow keeping Montse to take Don Quixote?  
**Only** if we remove from the DB the book that Montse already has on loan



ID	Name
2321323N	Pepe
7484849P	Paco
2342312Q	Kike
4848994J	Montse
...	...

Title	ISBN
Quixot	0011
BBDD	0022
Postres	1122
Hamlet	2829
...	...

# Example 5. Attributes Definition

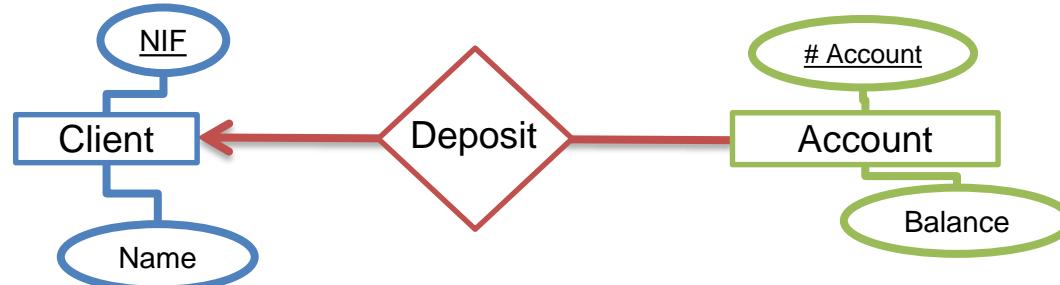
**Where is it more convenient to define an attribute?**

The cardinality of a relationship can determine where it is more convenient to define an attribute

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep access date for each bank account...



Client

Name	NIF
D. Alves	4748474P
R. Forns	6788900H
E. Walter	6112888L
R. Zabalo	7282944P
...	...

Deposit

Account

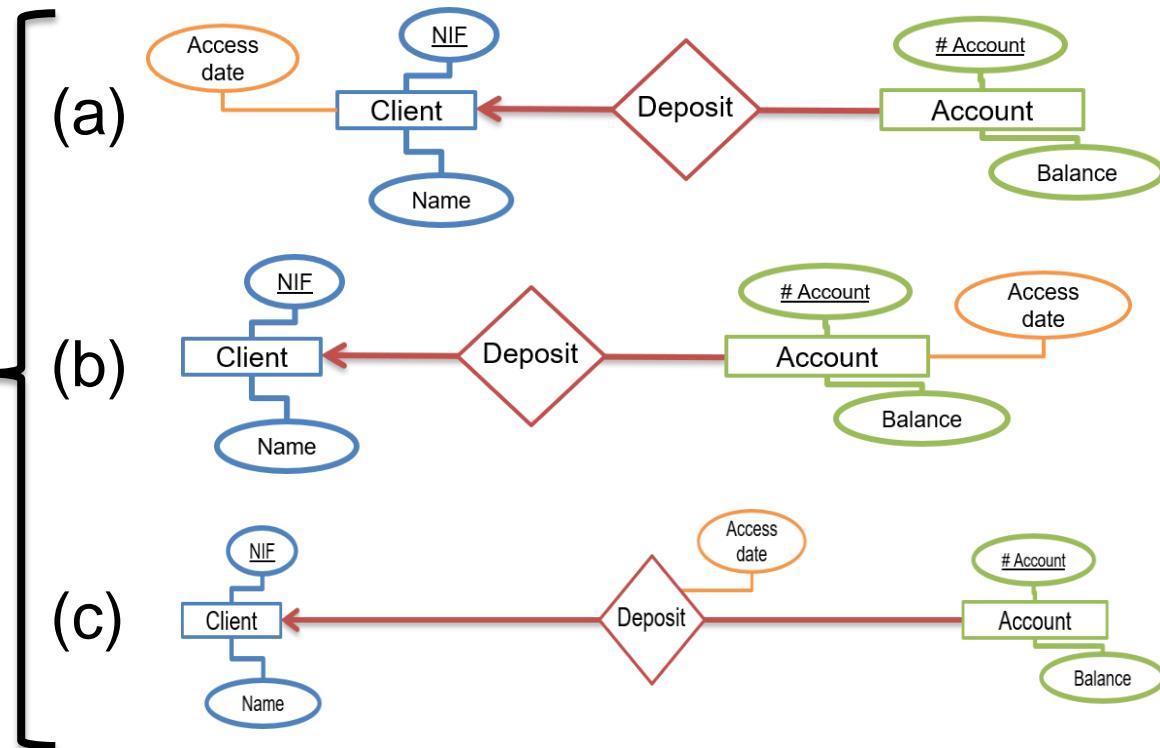
# Account	Balance
CTB-246800	500
CTB-357900	1.250
CTB-135700	13.000
CTB-975300	100.056
...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep **access date** for each bank account...

There are 3 options:-

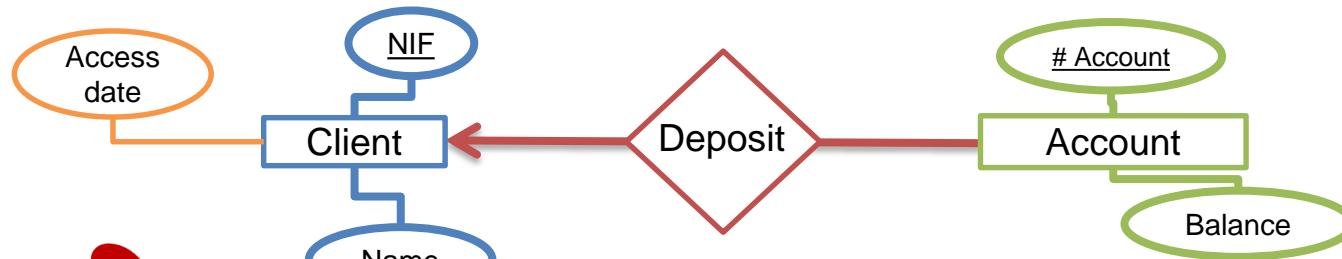


## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep access date for each bank account...

Option (a)



Client

Access date	Name	NIF
22-05-2016	D. Alves	4748474P
15-04-2016	R. Forns	6788900H
22-02-2017	E. Walter	6112888L
25-01-2017	R. Zabalo	7282944P
...	...	...

Deposit

Account

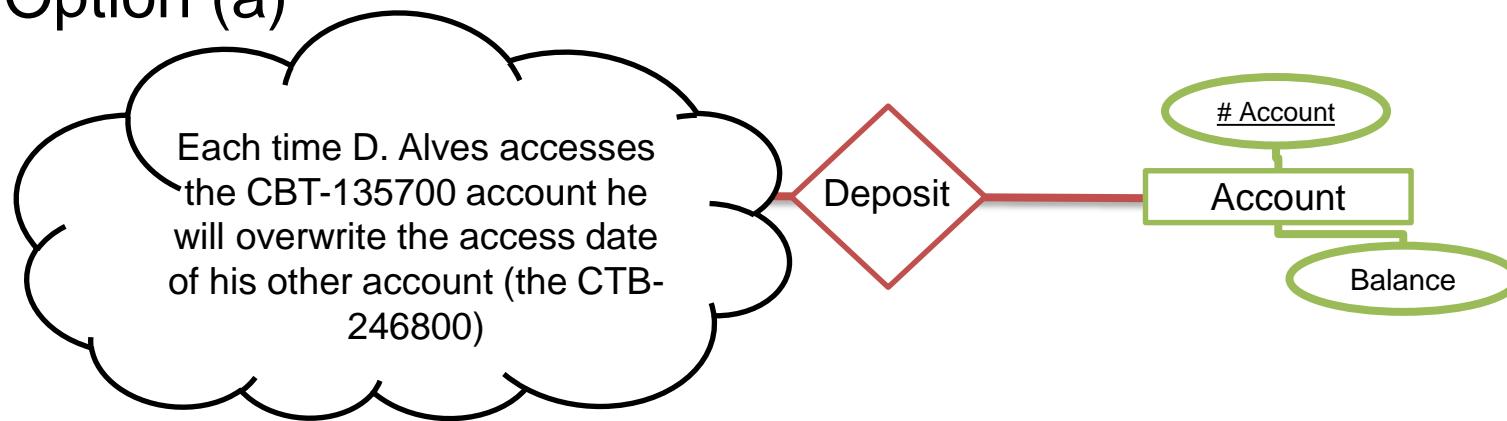
# Account	Balance
CTB-246800	500
CTB-357900	1.250
CTB-135700	13.000
CTB-975300	100.056
...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep access date for each bank account...

Option (a)



Access date	Name	NIF
22-05-2016	D. Alves	4748474P
15-04-2016	R. Forns	6788900H
22-02-2017	E. Walter	6112888L
25-01-2017	R. Zabalo	7282944P
...	...	...

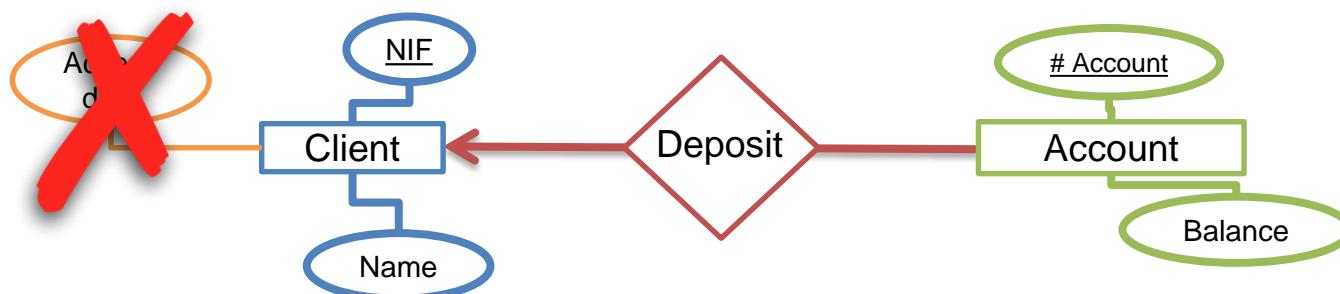
Deposit	
# Account	Balance
CTB-246800	500
CTB-357900	1.250
CTB-135700	13.000
CTB-975300	100.056
...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep access date for each bank account...

Option (a)



**Unable to record access dates**

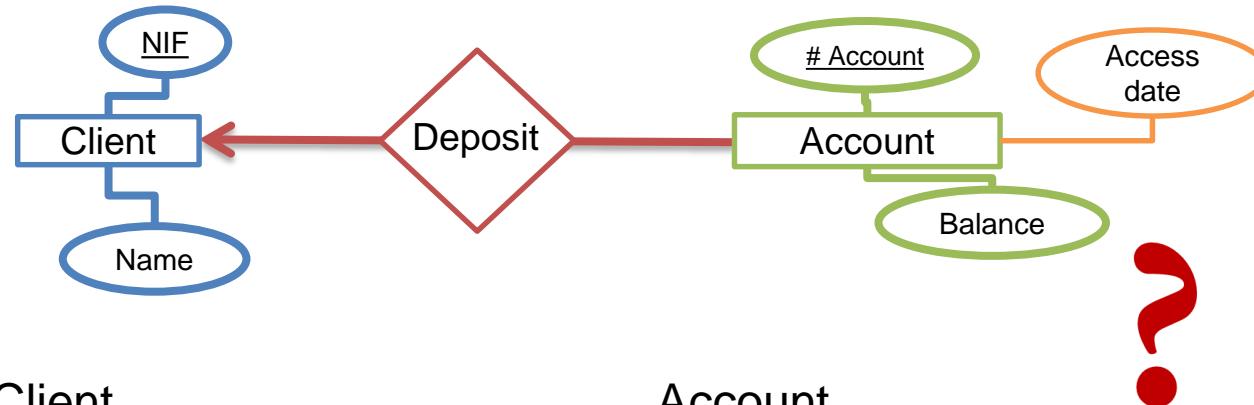
Access date	Name	NIF	Deposit	# Account	Balance
22-5-2016	D. Alves	4748474P		CTB-246800	500
15-6-2016	R. Forns	6788900H		CTB-357900	1.250
22-2-2017	E. Walter	6112888L		CTB-135700	13.000
5-01-2017	R. Zabalo	7282944P		CTB-975300	100.056
...	...	...		...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep access date for each bank account...

Option (b)



Client

Name	NIF
D. Alves	4748474P
R. Forns	6788900H
E. Walter	6112888L
R. Zabalo	7282944P
...	...

Deposit

Account

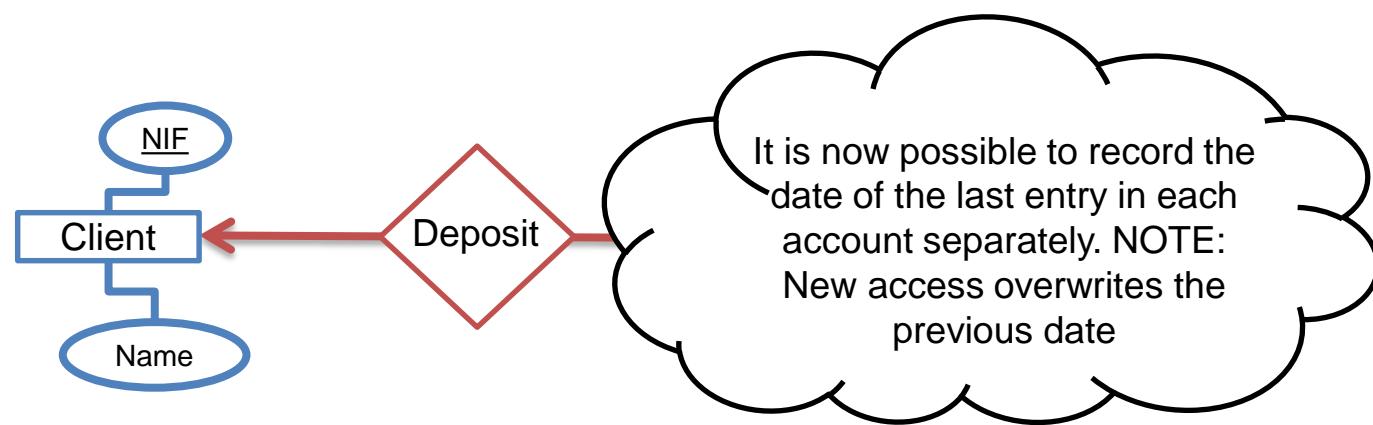
# Account	Balance	Access date
CTB-246800	500	22-05-2016
CTB-357900	1.250	15-04-2016
CTB-135700	13.000	22-02-2017
CTB-975300	100.056	25-01-2017
...	...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep access date for each bank account...

Option (b)



Client

Name	NIF
D. Alves	4748474P
R. Forns	6788900H
E. Walter	6112888L
R. Zabalo	7282944P
...	...

Deposit

Account

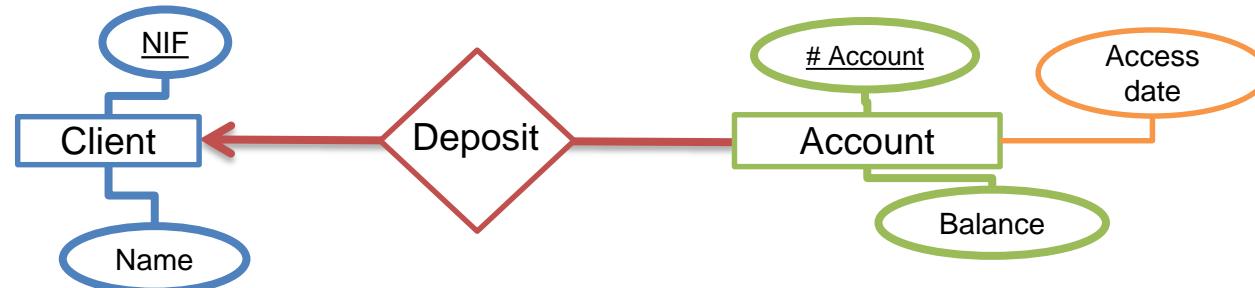
# Account	Balance	Access date
CTB-246800	500	22-05-2016
CTB-357900	1.250	15-04-2016
CTB-135700	13.000	22-02-2017
CTB-975300	100.056	25-01-2017
...	...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep access date for each bank account...

Option (b)



In this case you get a possible solution (last access date)

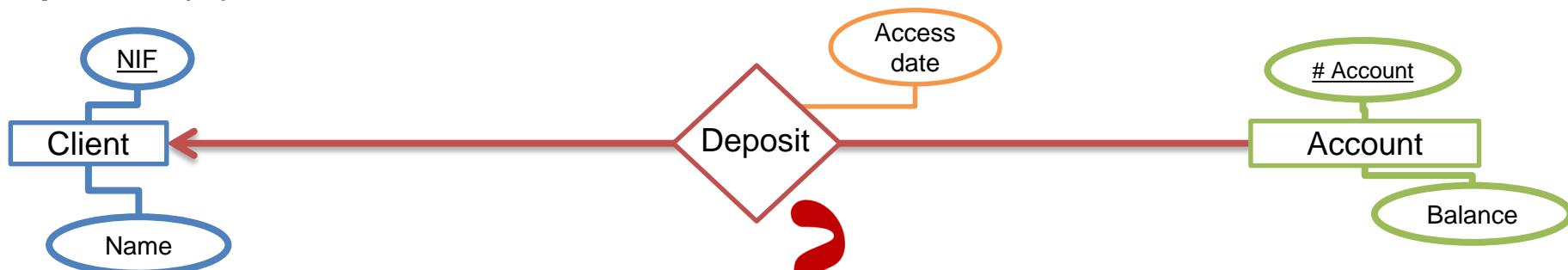
Name	NIF	Deposit	# Account	Balance	Access date
D. Alves	4748474P		CTB-246800	500	22-05-2016
R. Forns	6788900H		CTB-357900	1.250	15-04-2016
E. Walter	6112888L		CTB-135700	13.000	22-02-2017
R. Zabalo	7282944P		CTB-975300	100.056	25-01-2017
...	...		...	...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep access date for each bank account...

Option (c)



Client

Name	NIF
D. Alves	4748474P
R. Forns	6788900H
E. Walter	6112888L
R. Zabalo	7282944P
...	...

Deposit

Access date
22-05-2016
15-04-2016
22-02-2017
25-01-2017
...

Account

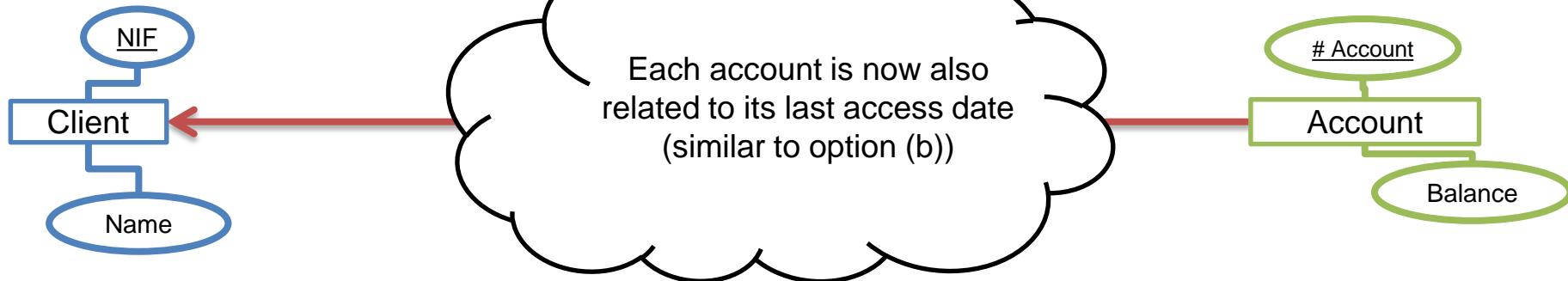
# Account	Balance
CTB-246800	500
CTB-357900	1.250
CTB-135700	13.000
CTB-975300	100.056
...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep access date for each bank account...

Option (c)



Client

Name	NIF
D. Alves	4748474P
R. Forns	6788900H
E. Walter	6112888L
R. Zabalo	7282944P
...	...

Deposit

Access date
22-05-2016
15-04-2016
22-02-2017
25-01-2017
...

Account

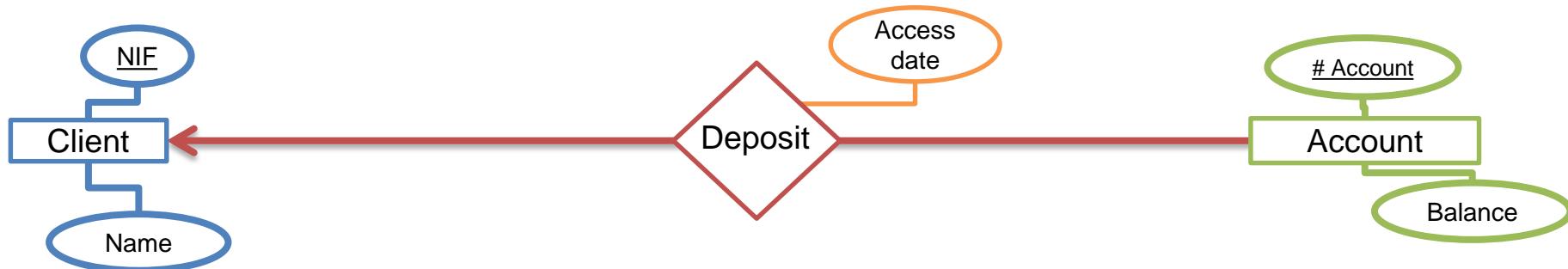
# Account	Balance
CTB-246800	500
CTB-357900	1.250
CTB-135700	13.000
CTB-975300	100.056
...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-N (one to many)

Suppose we want to keep access date for each bank account...

Option (c)



In this case you also get a possible solution (last access date)

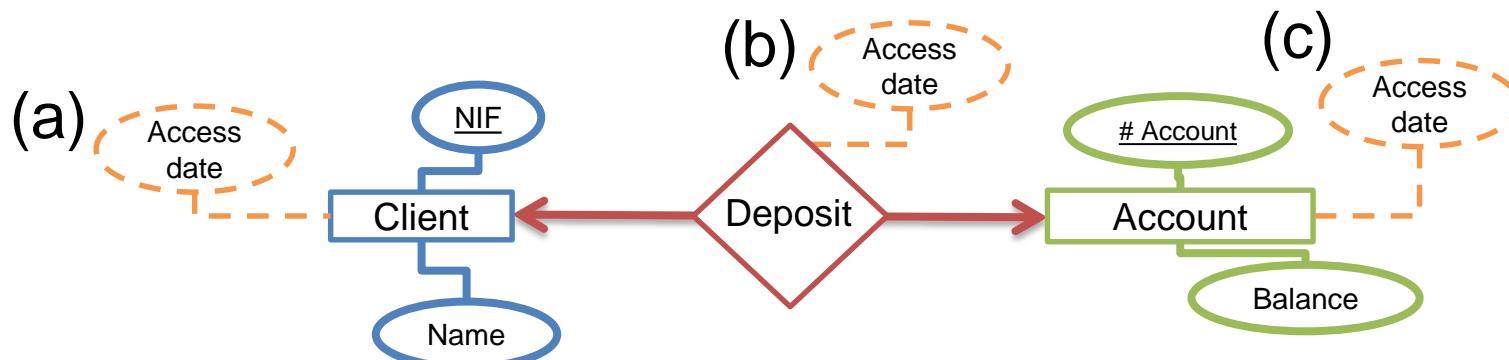
Name	NIF	Access date	# Account	Balance
D. Alves	4748474P	22-05-2016	CTB-246800	500
R. Forns	6788900H	15-04-2016	CTB-357900	1.250
E. Walter	6112888L	22-02-2017	CTB-135700	13.000
R. Zabalo	7282944P	25-01-2017	CTB-975300	100.056
...	...	...	...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-1 (one to one)

Suppose we want to keep access date for each bank account...

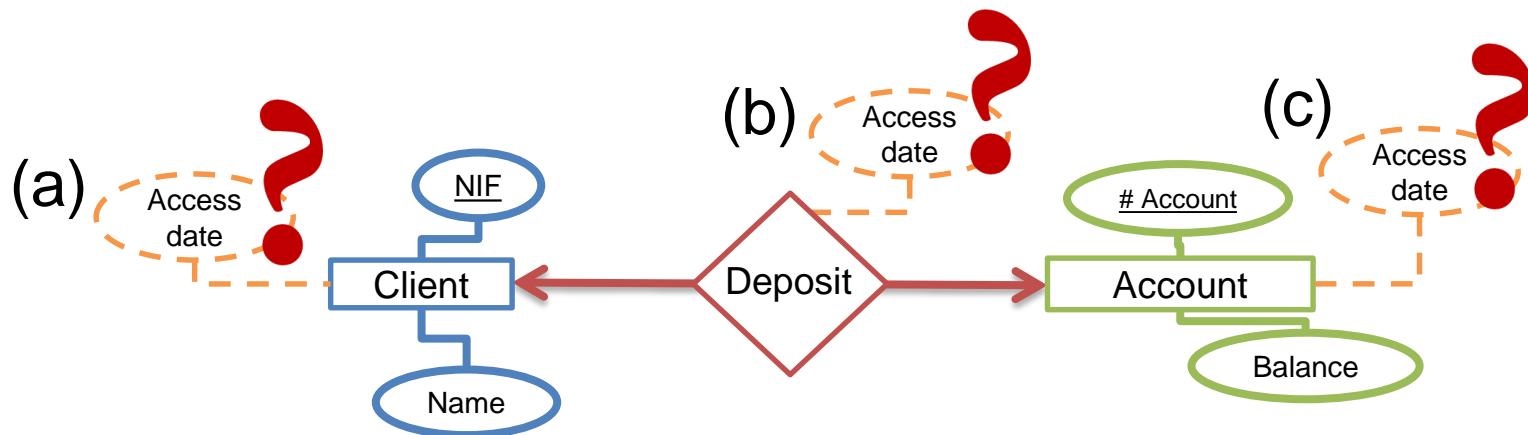
**Again there are three possibilities:**



## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-1 (one to one)

Suppose we want to keep access date for each bank account...



Client

Name	NIF
D. Alves	4748474P
R. Forns	6788900H
E. Walter	6112888L
R. Zabalo	7282944P
...	...

Deposit

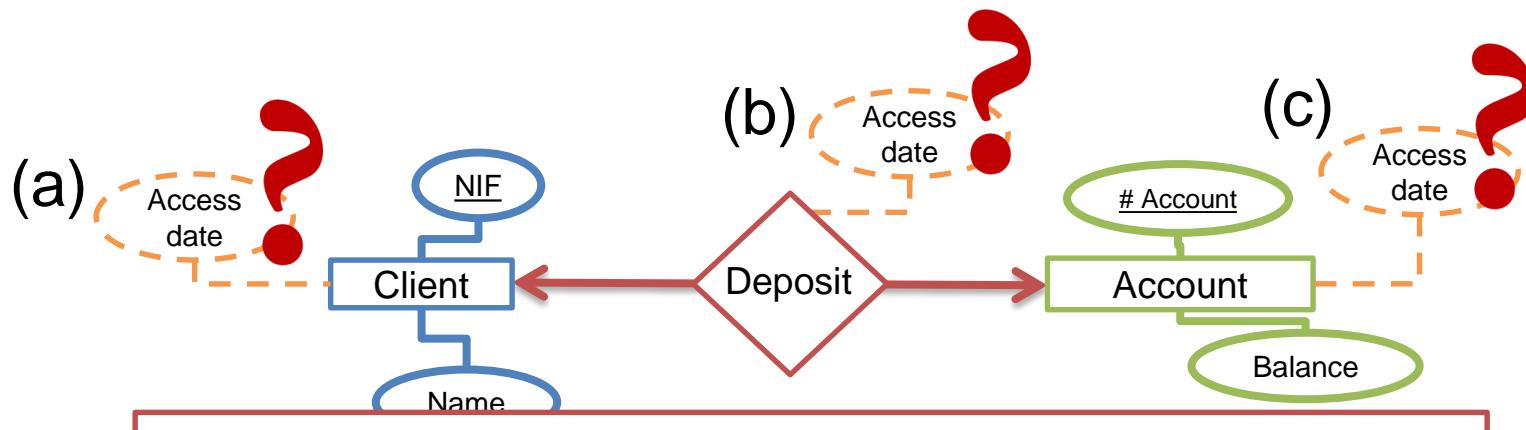
Account

# Account	Balance
CTB-246800	500
CTB-357900	1.250
CTB-135700	13.000
CTB-975300	100.056
...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is 1-1 (one to one)

Suppose we want to keep access date for each bank account...



In this case, all three designs retain the last access date for each account and are equivalent

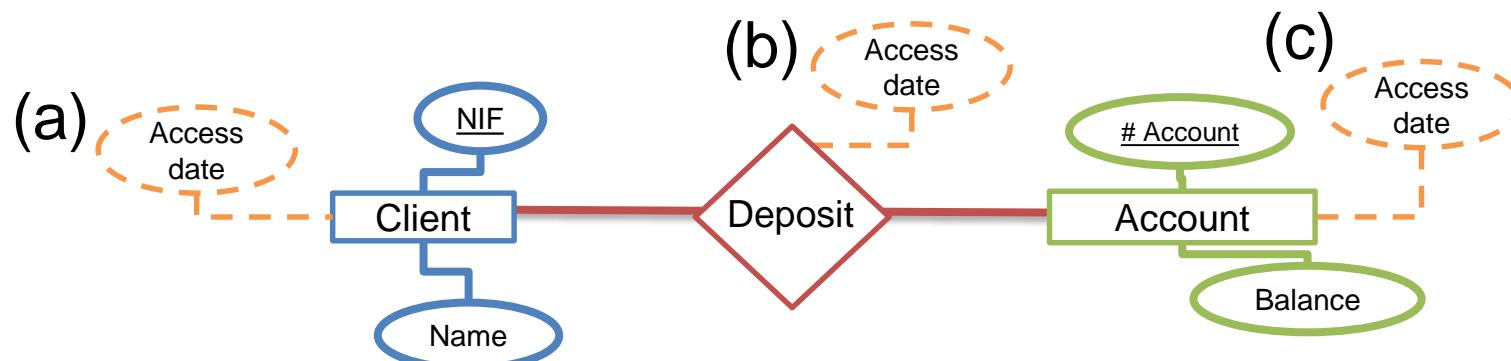
Deposit				
D. Alves	4748474P	CTB-246800	500	
R. Forns	6788900H	CTB-357900	1.250	
E. Walter	6112888L	CTB-135700	13.000	
R. Zabalo	7282944P	CTB-975300	100.056	
...	...	...	...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is **N-N** (many to many)

Suppose we want to keep access date for each bank account...

Again there are three possibilities:

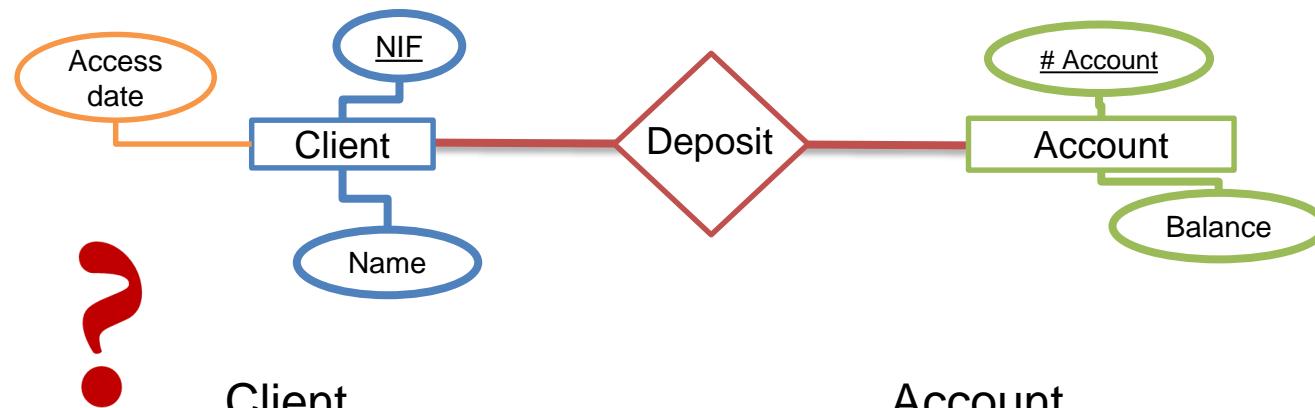


## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is N-N (many to many)

Suppose we want to keep access date for each bank account...

Option (a)



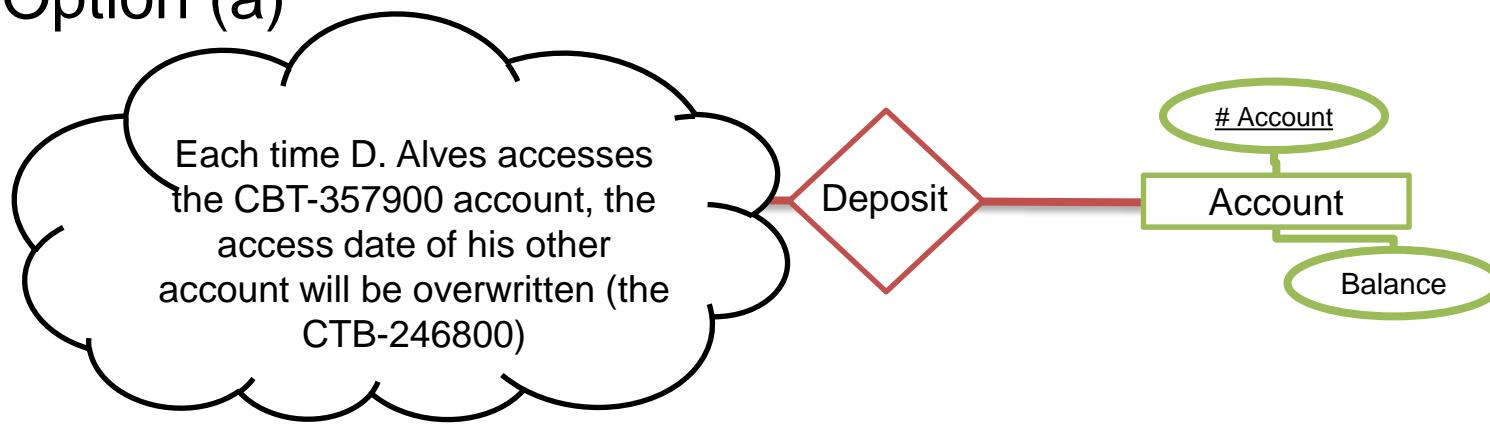
Client			Deposit		Account	
Access date	Name	NIF			# Account	Balance
22-05-2016	D. Alves	4748474P			CTB-246800	500
	R. Forns	6788900H			CTB-357900	1.250
22-02-2017	E. Walter	6112888L			CTB-135700	13.000
25-01-2017	R. Zabalo	7282944P			CTB-975300	100.056
...	...	...			...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is N-N (many to many)

Suppose we want to keep access date for each bank account...

Option (a)



Client			# Account	Balance
Access date	Name	NIF		
22-05-2016	D. Alves	4748474P	CTB-246800	500
	R. Forns	6788900H	CTB-357900	1.250
22-02-2017	E. Walter	6112888L	CTB-135700	13.000
25-01-2017	R. Zabalo	7282944P	CTB-975300	100.056
...	...	...	...	...

**Deposit**

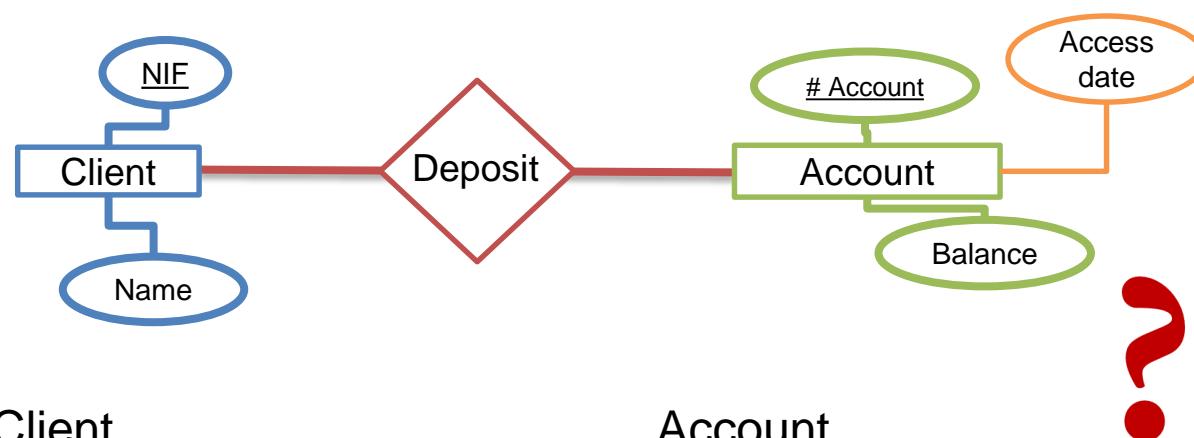
# Account	Balance
CTB-246800	500
CTB-357900	1.250
CTB-135700	13.000
CTB-975300	100.056
...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is N-N (many to many)

Suppose we want to keep access date for each bank account...

Option (b)



Client

Name	NIF
D. Alves	4748474P
R. Forns	6788900H
E. Walter	6112888L
R. Zabalo	7282944P
...	...

Deposit

Account

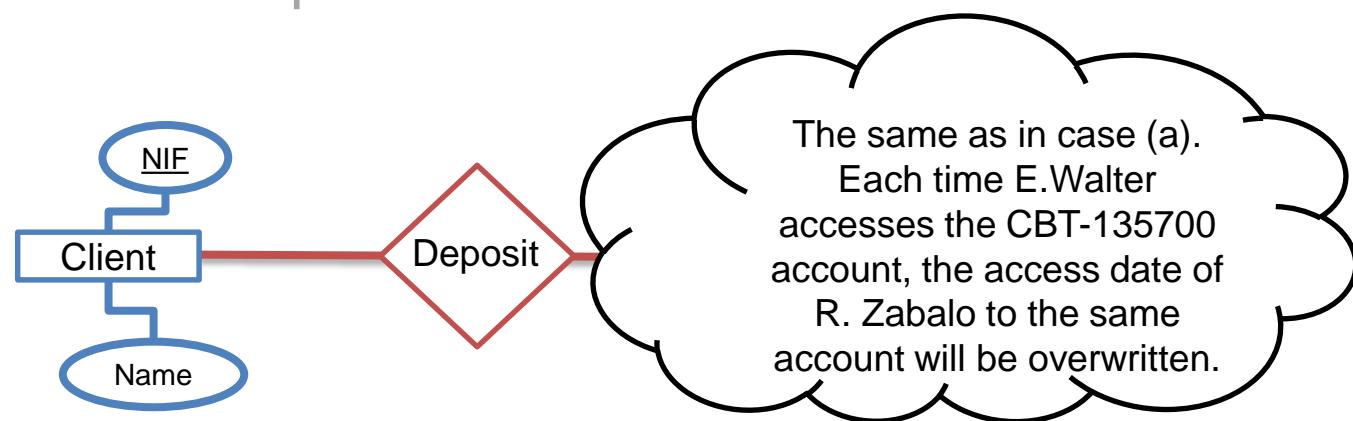
# Account	Balance	Access date
CTB-246800	500	22-05-2016
CTB-357900	1.250	15-04-2016
CTB-135700	13.000	22-02-2017
CTB-975300	100.056	25-01-2017
...	...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is N-N (many to many)

Suppose we want to keep access date for each bank account...

Option (b)



Client

Name	NIF
D. Alves	4748474P
R. Forns	6788900H
<b>E. Walter</b>	6112888L
<b>R. Zabalo</b>	7282944P
...	...

Deposit

Account

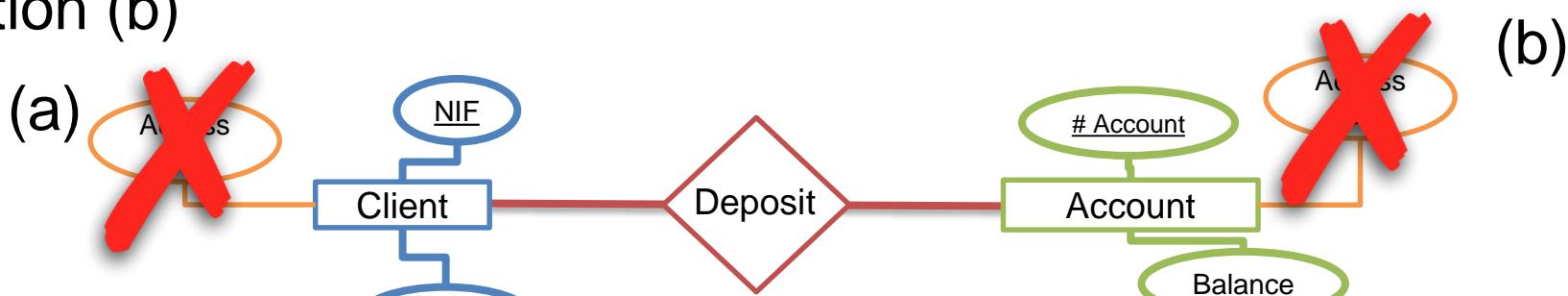
# Account	Balance	Access date
CTB-246800	500	22-05-2016
CTB-357900	1.250	15-04-2016
<b>CTB-135700</b>	13.000	22-02-2017
CTB-975300	100.056	25-01-2017
...	...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is N-N (many to many)

Suppose we want to keep access date for each bank account...

Option (b)



In no case (a) and (b) it is possible to register the clients and the dates of access to their respective accounts

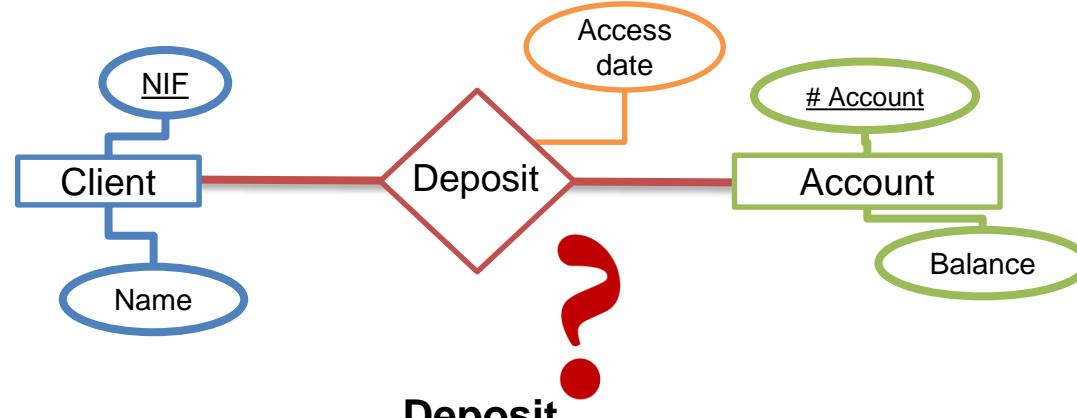
Access date	Name	NIF	Deposit	# Account	Balance	Access date
22-5-2016	D. Alves	4748474P		CTB-246800	500	22-5-2016
	R. Forns	6788900H		CTB-357900	1.250	15-6-2016
22-2-2017	E. Walter	6112888L		CTB-135700	13.000	22-2-2017
0-01-2017	R. Zabalo	7282944P		CTB-975300	100.056	0-01-2017
...	...	...		...	...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is N-N (many to many)

Suppose we want to keep access date for each bank account...

Option (c)



Client

Name	NIF
D. Alves	4748474P
R. Forns	6788900H
E. Walter	6112888L
R. Zabalo	7282944P
...	...

Deposit

Access date
22-05-2016
15-04-2016
22-02-2017
25-01-2017
...

Account

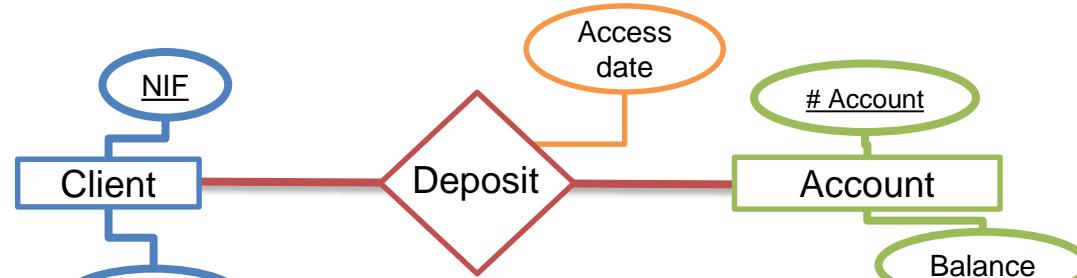
# Account	Balance
CTB-246800	500
CTB-357900	1.250
CTB-135700	13.000
CTB-975300	100.056
...	...

## Example 5. Attributes definition

Consider the E-R design where the cardinality of the “deposit” relationship is N-N (many to many)

Suppose we want to keep access date for each bank account...

Option (c)



This is the only case where it is possible to register the clients and the last dates of access to their respective accounts

Name	NIF	Access date	# Account	Balance
D. Alves	4748474P	22-05-2016	CTB-246800	500
R. Forns	6788900H	15-04-2016	CTB-357900	1.250
E. Walter	6112888L	22-02-2017	CTB-135700	13.000
R. Zabalo	7282944P	25-01-2017	CTB-975300	100.056
...	...	...	...	...

## Example 5. Attributes definition

**Final Exam Question:** Is it possible for these designs to record a "history" of clients that includes all access dates to their respective accounts?

# Block 2

# BASIC E/R DESIGN

## (PART 3)

Debora Gil, Oriol Ramos, Alejandro Párraga

# Basic Design ER Contents

ER Model 1. Introduction

2. Basic Structures

    2.1 Entities

    2.2 Attributes

    2.3 Relationships

3. Relationships Features

    3.1 Cardinality

    3.2 Degree

    3.3 Participation

# Basic Design ER Contents

ER Model 1. Introduction

## 2. Basic Structures

2.1 Entities

2.2 Attributes

2.3 Relationships

## 3. Relationships Features

3.1 Cardinality

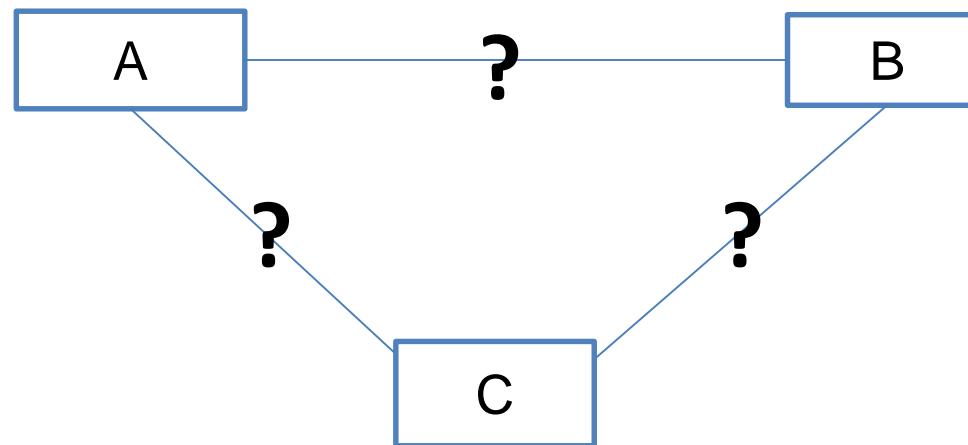
3.2 Degree

3.3 Participation

# Conceptual Model: Entity-Relationship Design

## Multiple relationships:

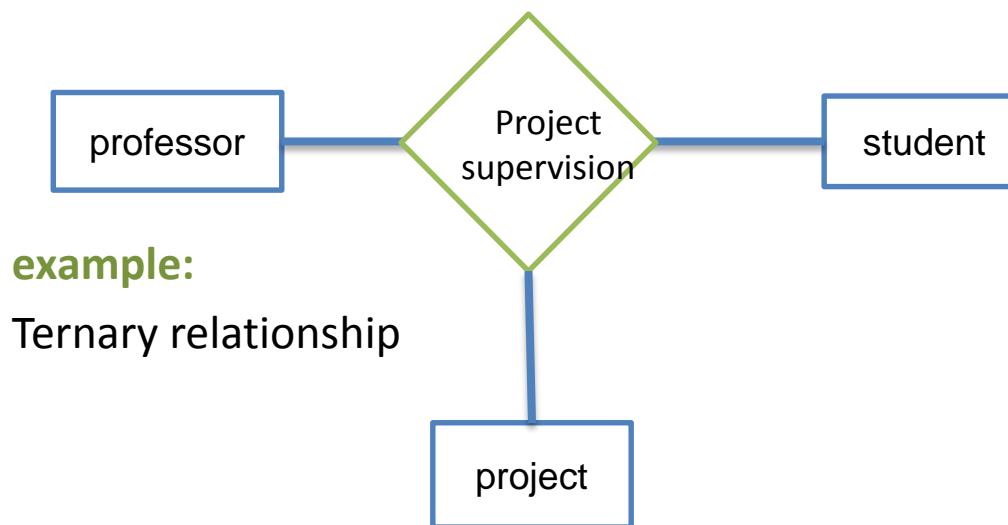
While most relationships are binary, there are times when it is more convenient to associate **elements from more than one entity**



# Conceptual Model: Entity-Relationship Design

## Multiple relationships:

While most relationships are binary, there are times when it is more convenient to associate **elements from more than one entity**



# Conceptual Model: Entity-Relationship Design

## Type of relationship:

As the number of entities involved, a relationship can be classified as:

- **unary (1-degree)**: relation to the same entity (hierarchical structure or equivalent structure).
- **Binary (2-degree)**: relationship between two entities. It is the most common.
- **Ternary (3-degree)**: relationship between the three entities.
- **n-ary (n-degree)**: relationship between n (known) entities.

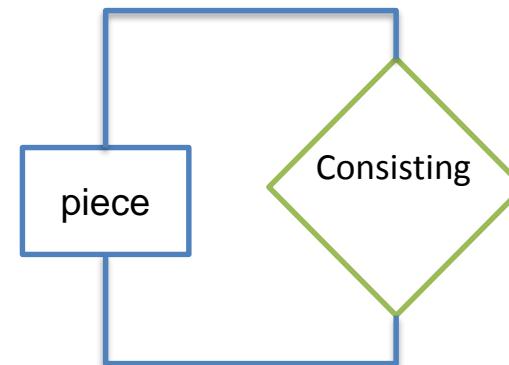
# Conceptual Model: Entity-Relationship Design

## Type of relationship:

- **unary (1-degree):** Relation of an entity with itself (Hierarchical)

A piece can be composed of other pieces.

**example:** Hinge comprising a bis and a lid.



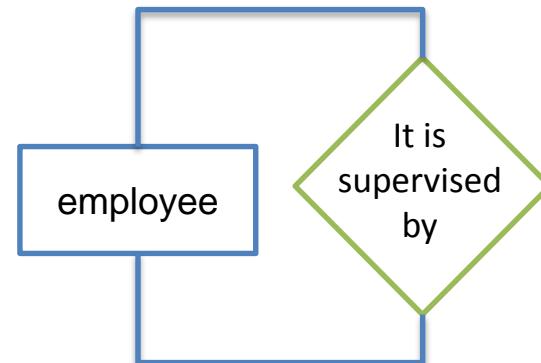
# Conceptual Model: Entity-Relationship Design

## Type of relationship:

- **unary (1-degree):** Relation of an entity with itself (Hierarchical)

A worker can work under the supervision of other employees.

**example:** an employee of a bank can be supervised by a manager (another employee of the same bank).



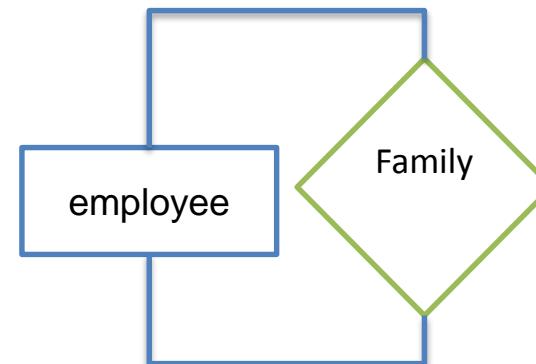
# Conceptual Model: Entity-Relationship Design

## Type of relationship:

- **unary (1-degree):** Relation of an entity with itself (Equivalent structure)

Unary equivalent relationships no establish hierarchy

**example:** A worker can be family of another worker (and this for the first).



# Conceptual Model: Entity-Relationship Design

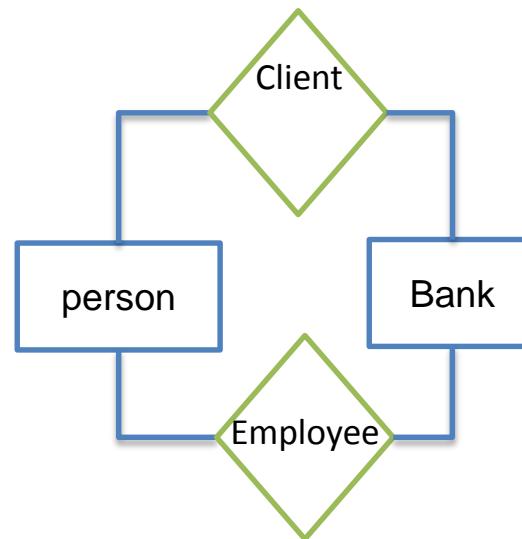
## Type of relationship:

- **Binary Relationship (2-degree):** relationship between two entities. The most common.

### Multiple binary relations:

It may be the case of more than one binary relation between two entities.

**example:** An employee can be,also, a bank client



# Conceptual Model: Entity-Relationship Design

## Type of relationship:

- **Ternary relationship (3-degree):** relationship among three entities.

**It is usually possible** replace a relations of degree > 2 by a number of different binary relations.

In some cases these two designs are not equivalent

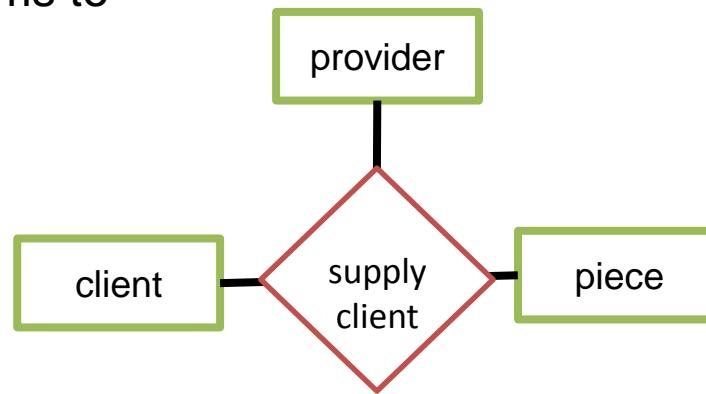
They are very rigid structures

# Conceptual Model: Entity-Relationship Design

## Type of relationship:

- **Ternary relationship (3-degree):** relationship among three entities.

**example:** Some vendors provide certain items to certain customers

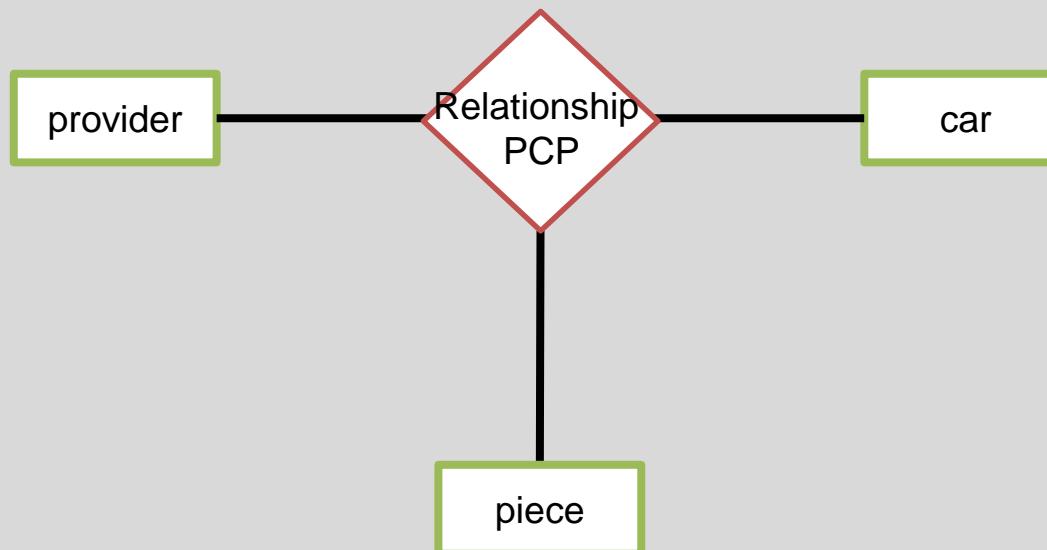


# Ternary Relationships cardinality (degree 3)

## Discussion 2: Ternary relationships cardinality

Consider the following ER design.

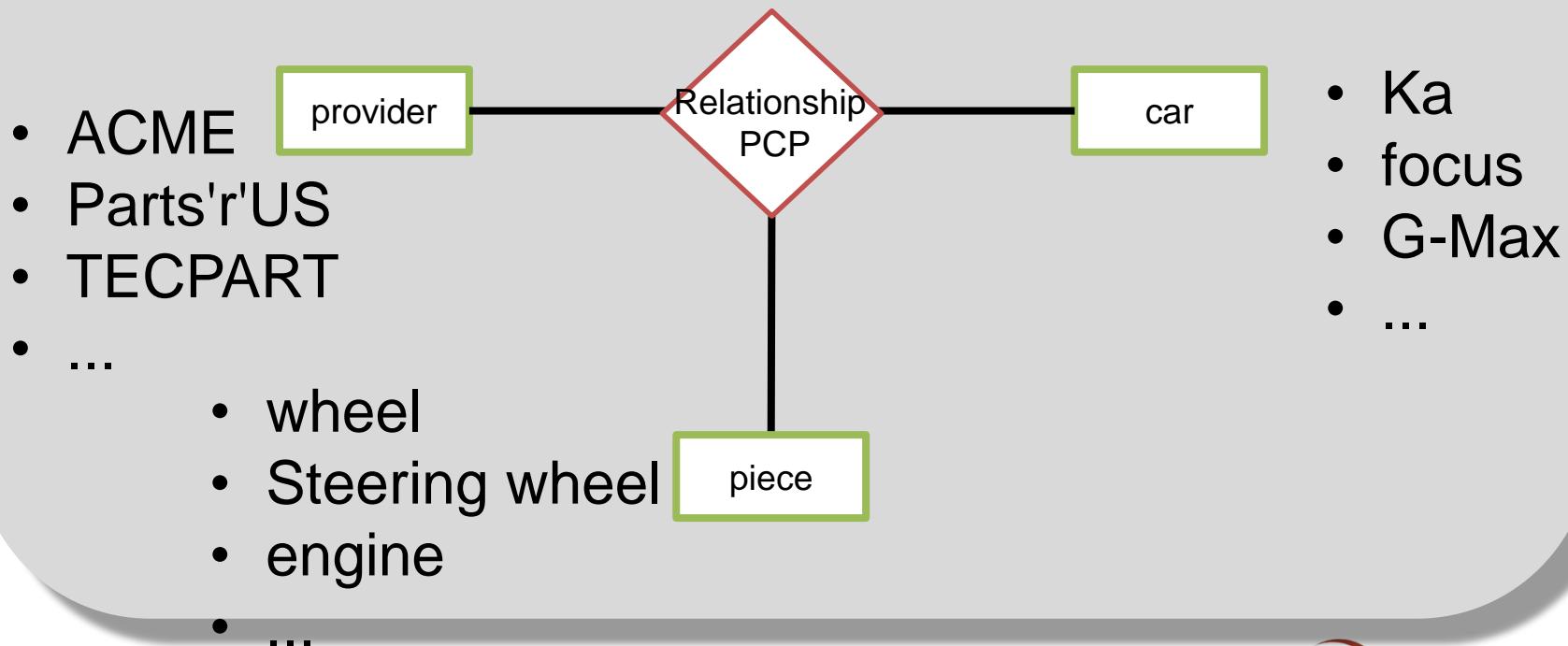
Expressing cardinalities.



# Reviewing Cardinality

## Discussion 2: Ternary relationship cardinality

Consider the following instances

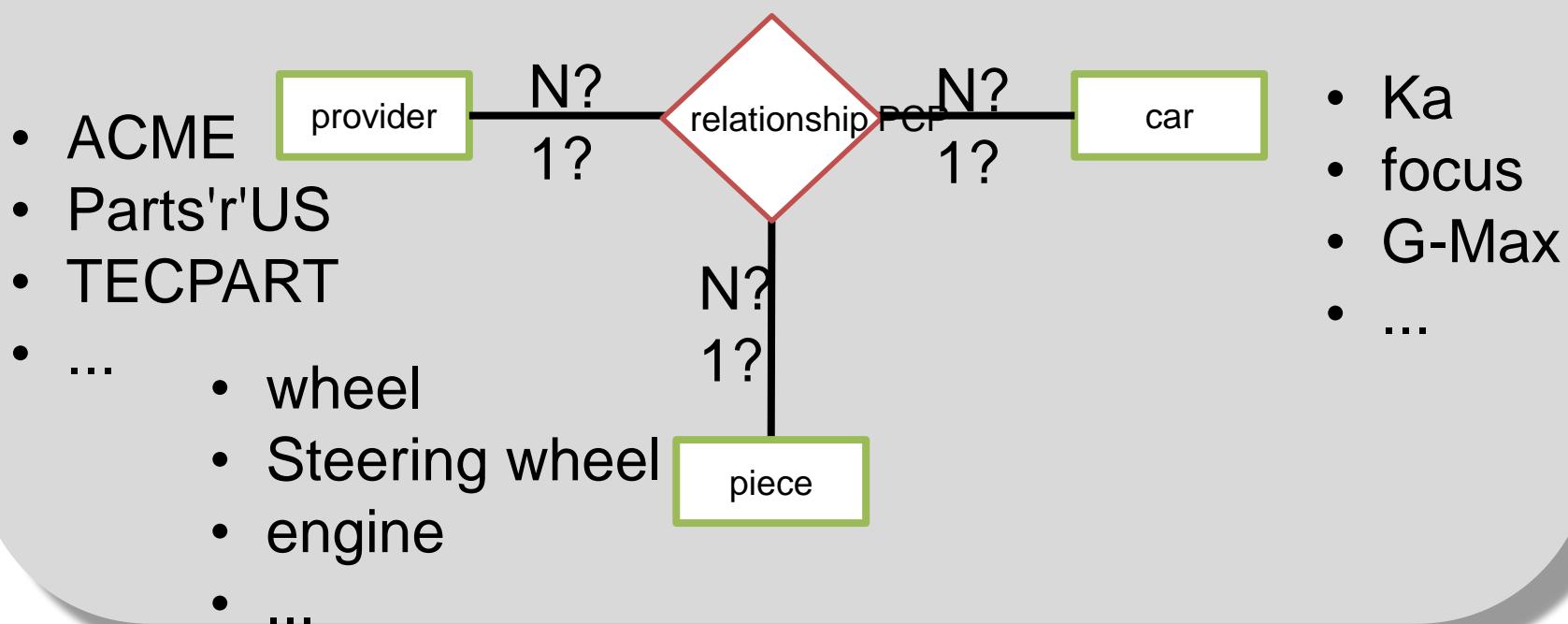


# Reviewing Cardinality

## Discussion 2: Ternary relationships cardinality

Expressing cardinalities.

Consider the following ...

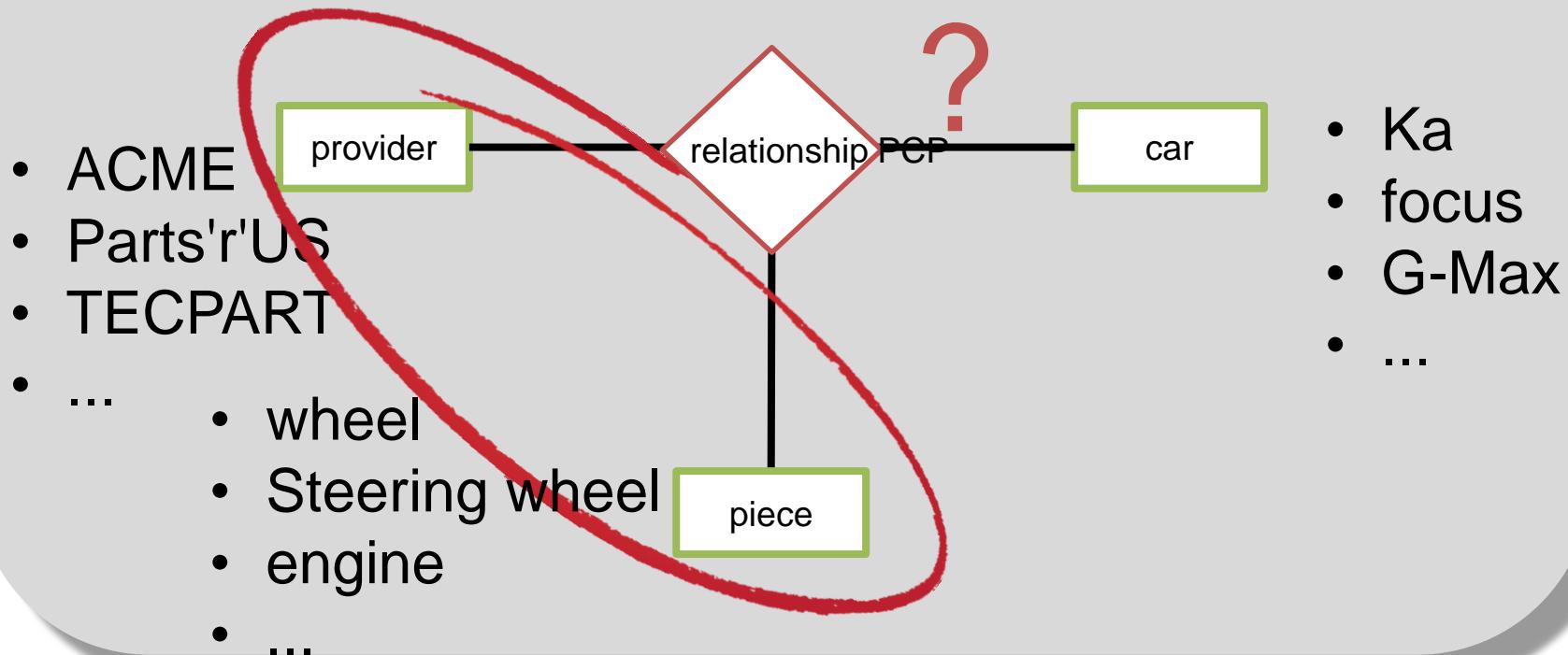


# Reviewing Cardinality

## Discussion 2:

For a given instance of a given supplier of a given piece .... Can this particular combination (Provider / piece) satisfies several cars or at most just a car?

?

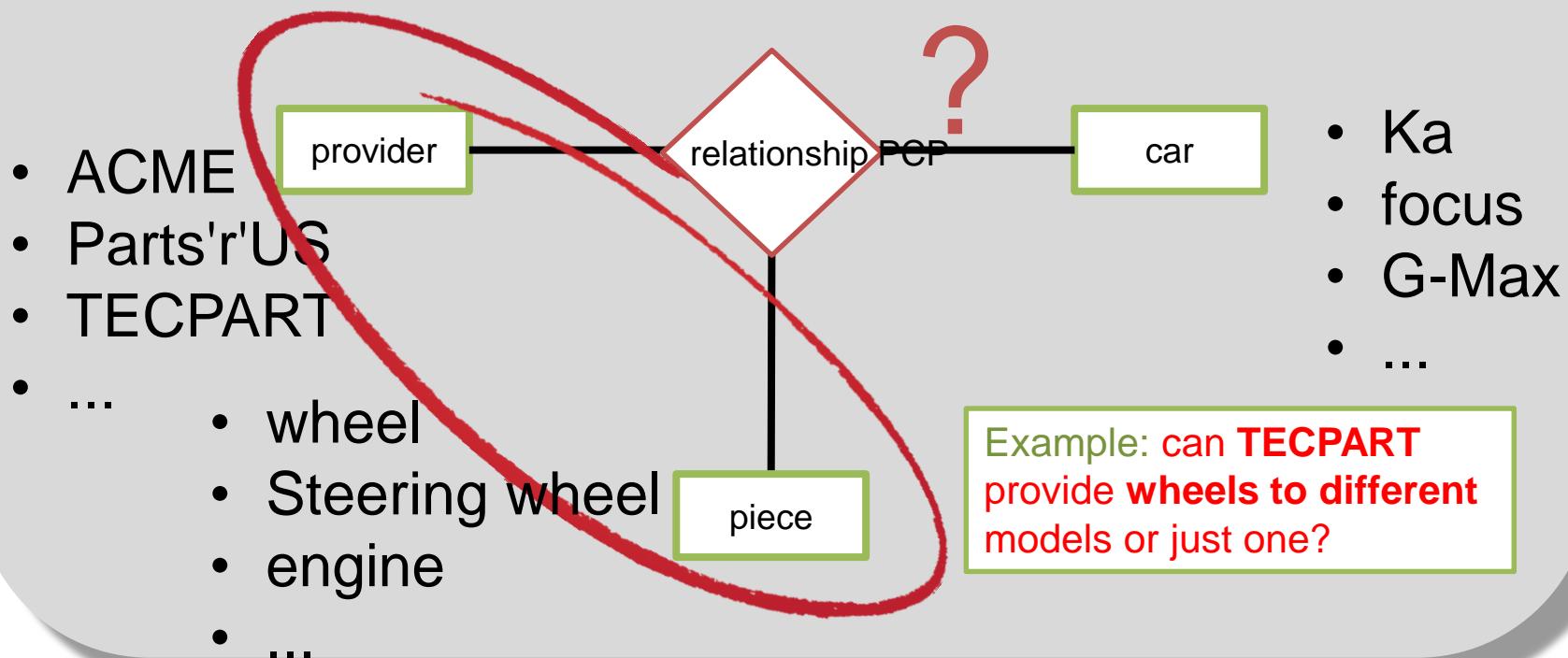


# Reviewing Cardinality

## Discussion 2:

For a given instance of a given supplier of a given piece .... Can this particular combination (Provider / piece) satisfies several cars or at most just a car?

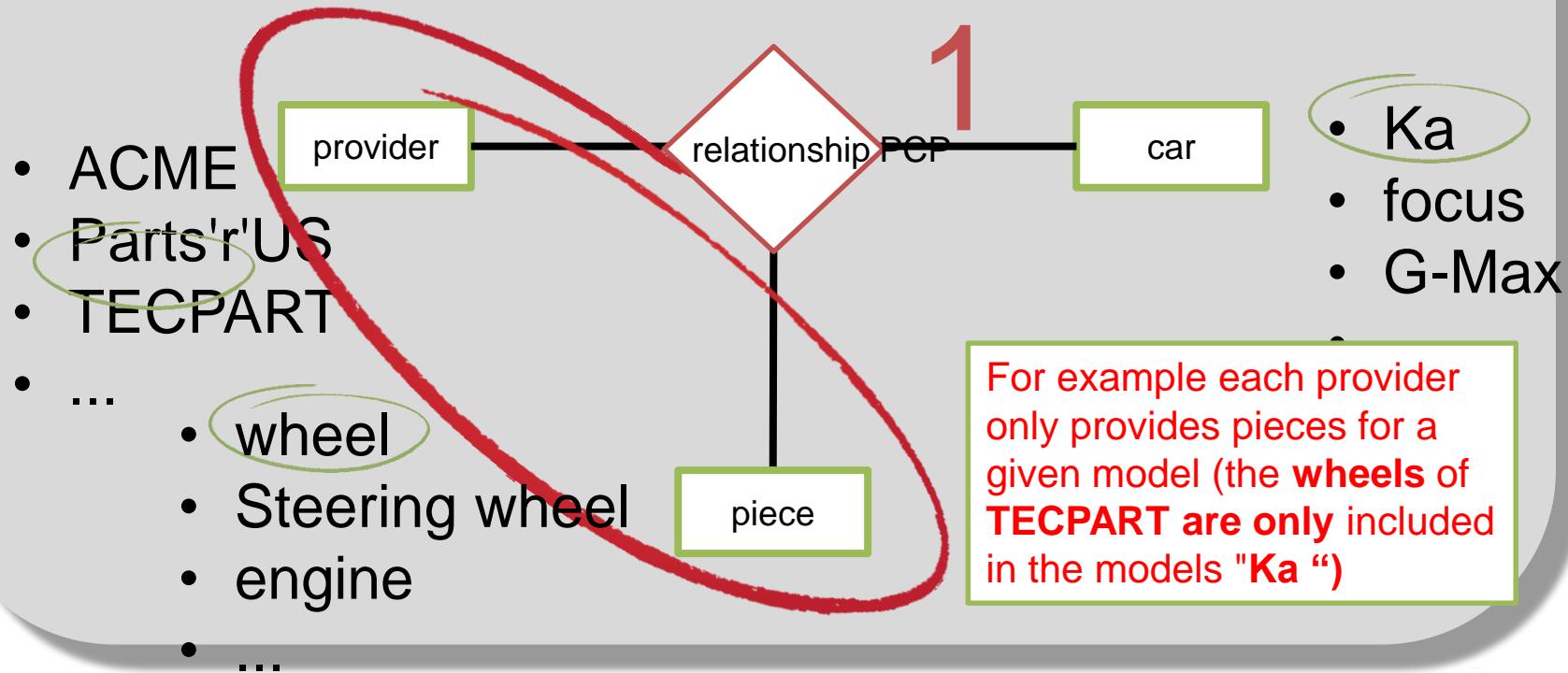
?



# Reviewing Cardinality

## Discussion 2:

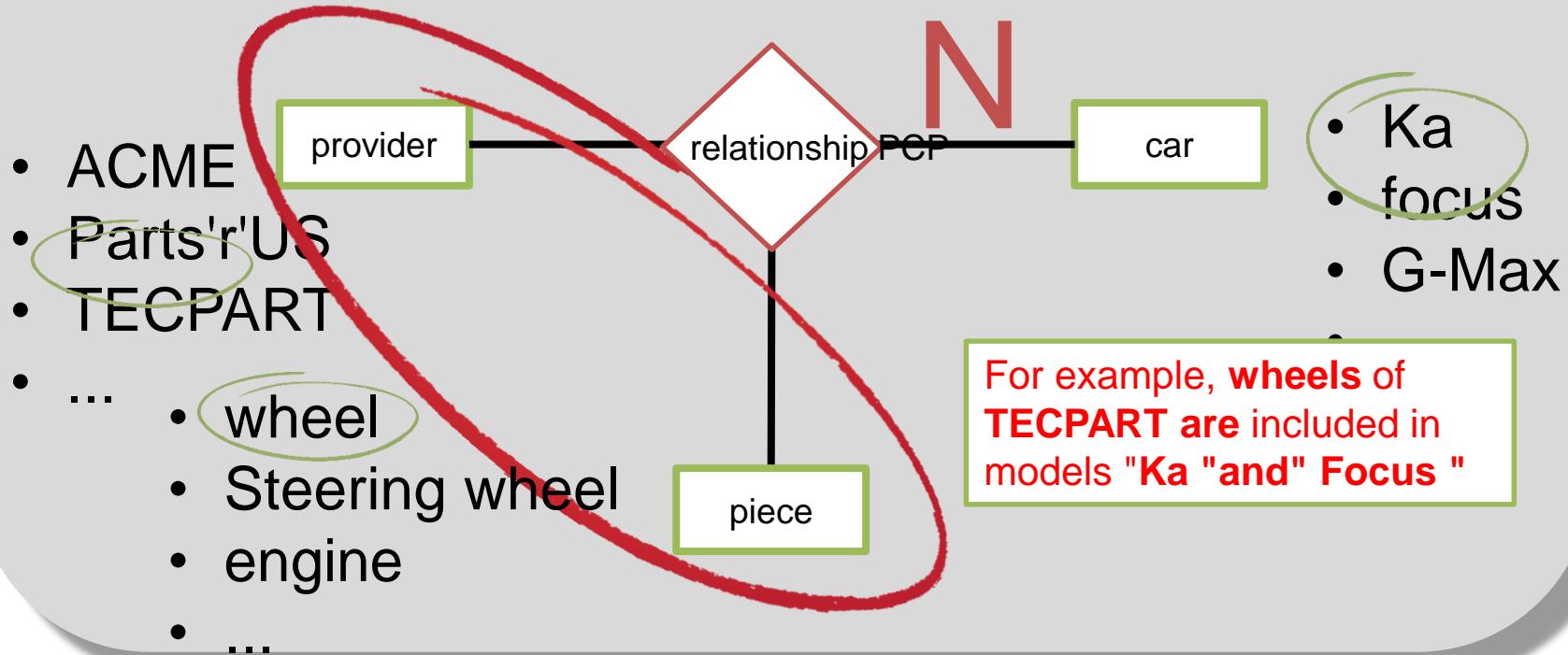
If combinations of provider and piece can only meet one model car, then the car cardinality is 1



# Reviewing Cardinality

## Discussion 2:

If a combination of provider and piece can satisfy several cars, then the “car” cardinality is N (many)

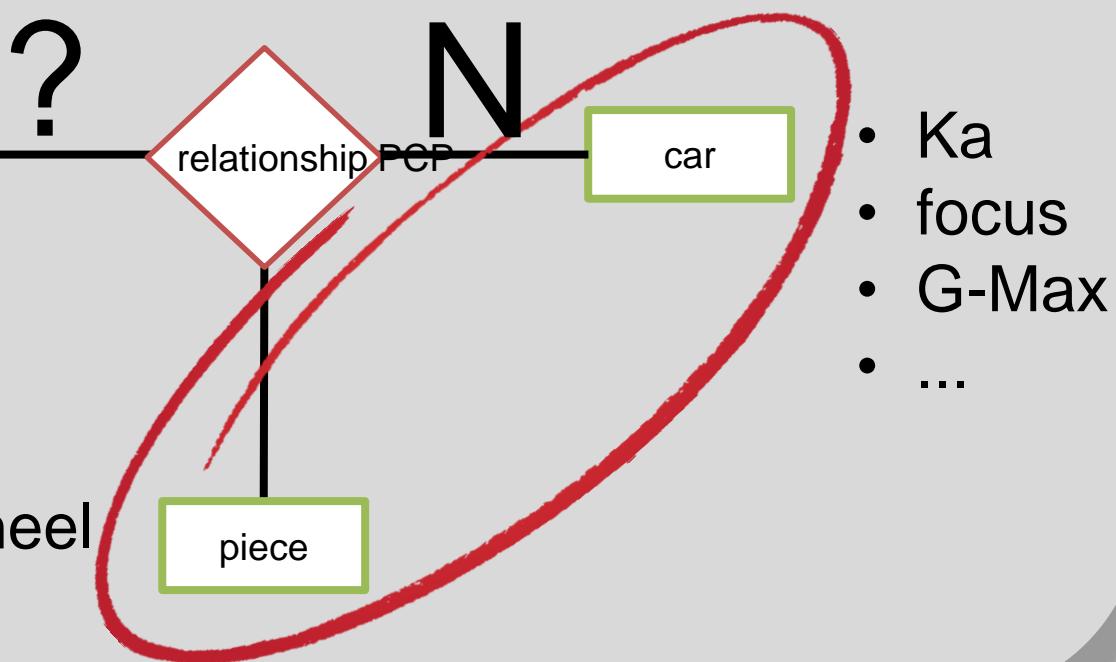


# Reviewing Cardinality

## Discussion 2:

Same for the other combinations: for a given instance of a car and a given piece .... Can this particular combination (car / piece) relate to several suppliers or simply with a single supplier at most?

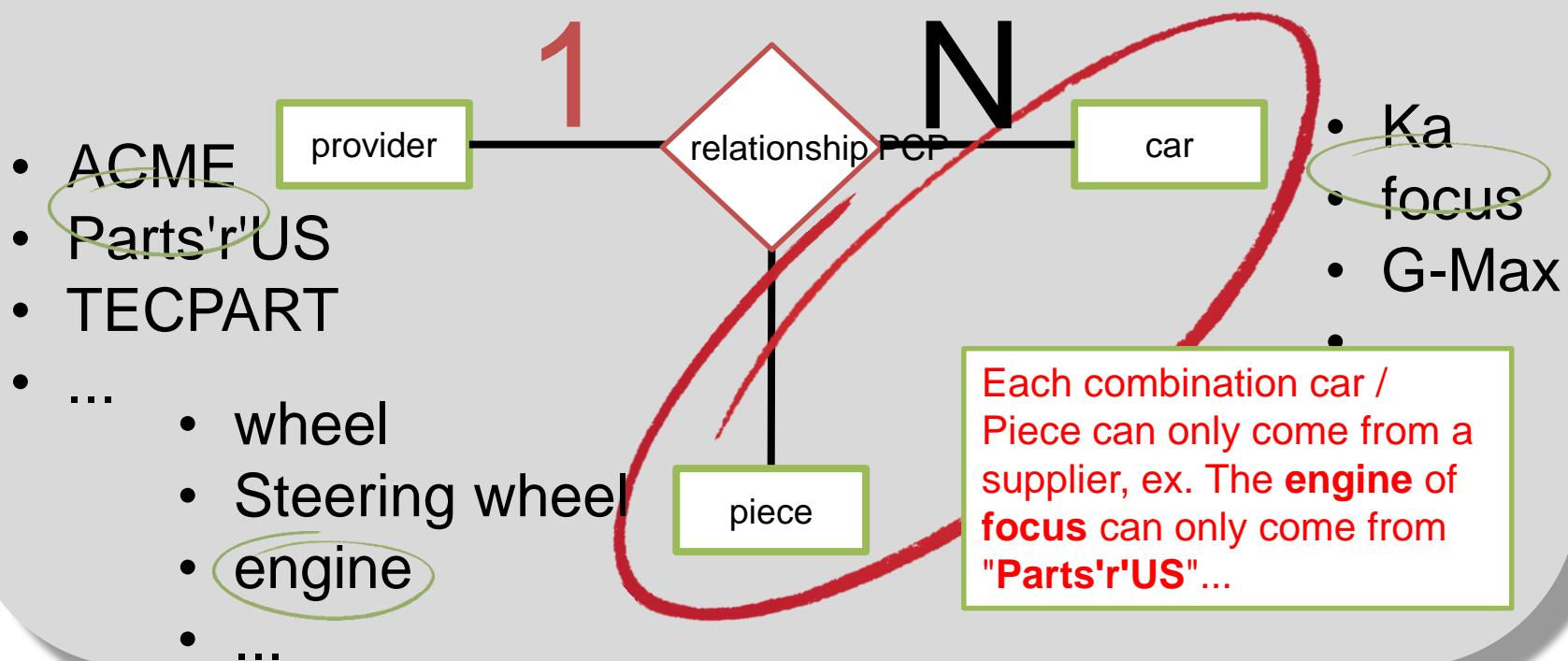
- ACME
- Parts'r'US
- TECPART
- ...
  - wheel
  - Steering wheel
  - engine
  - ...



# Reviewing Cardinality

## Discussion 2:

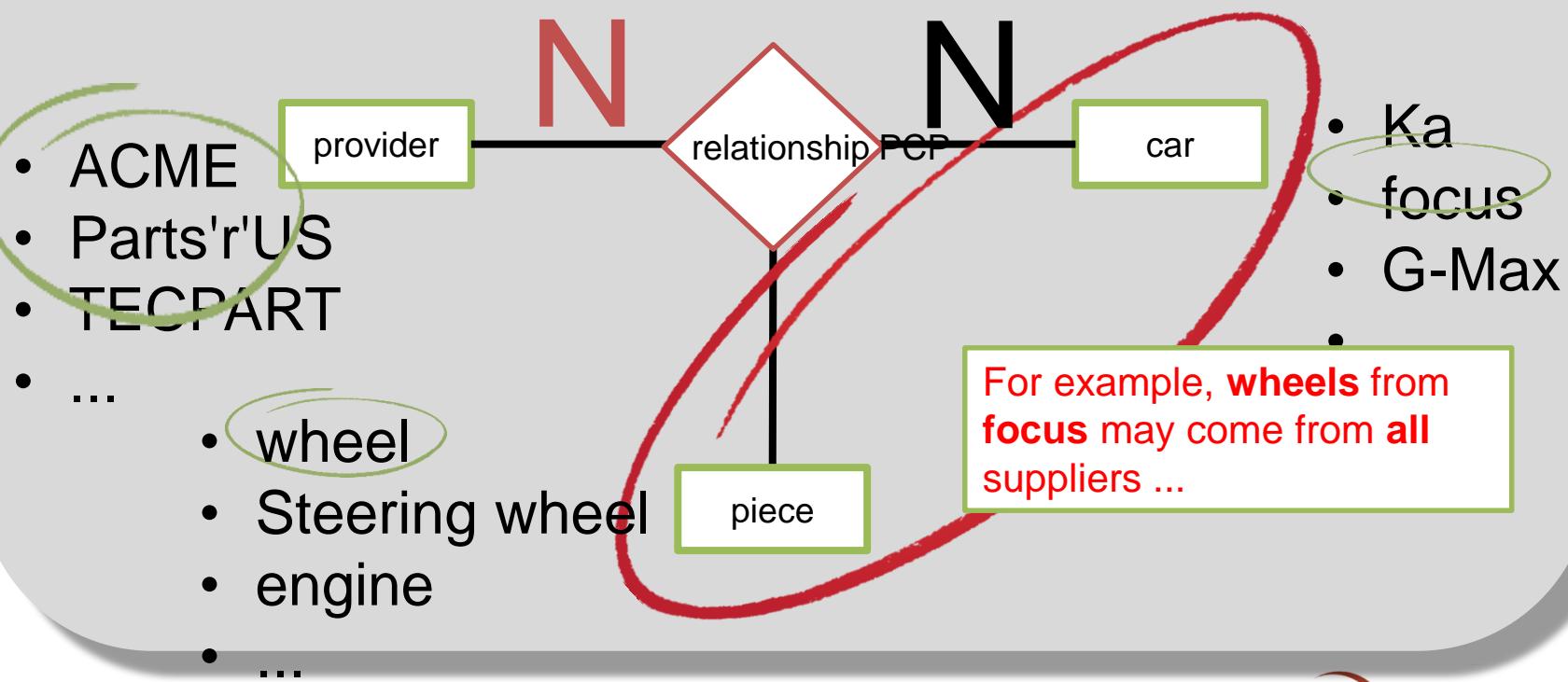
If a combination car-piece can only come from a single provider, then the cardinality of "Provider" is 1.



# Reviewing Cardinality

## Discussion 2:

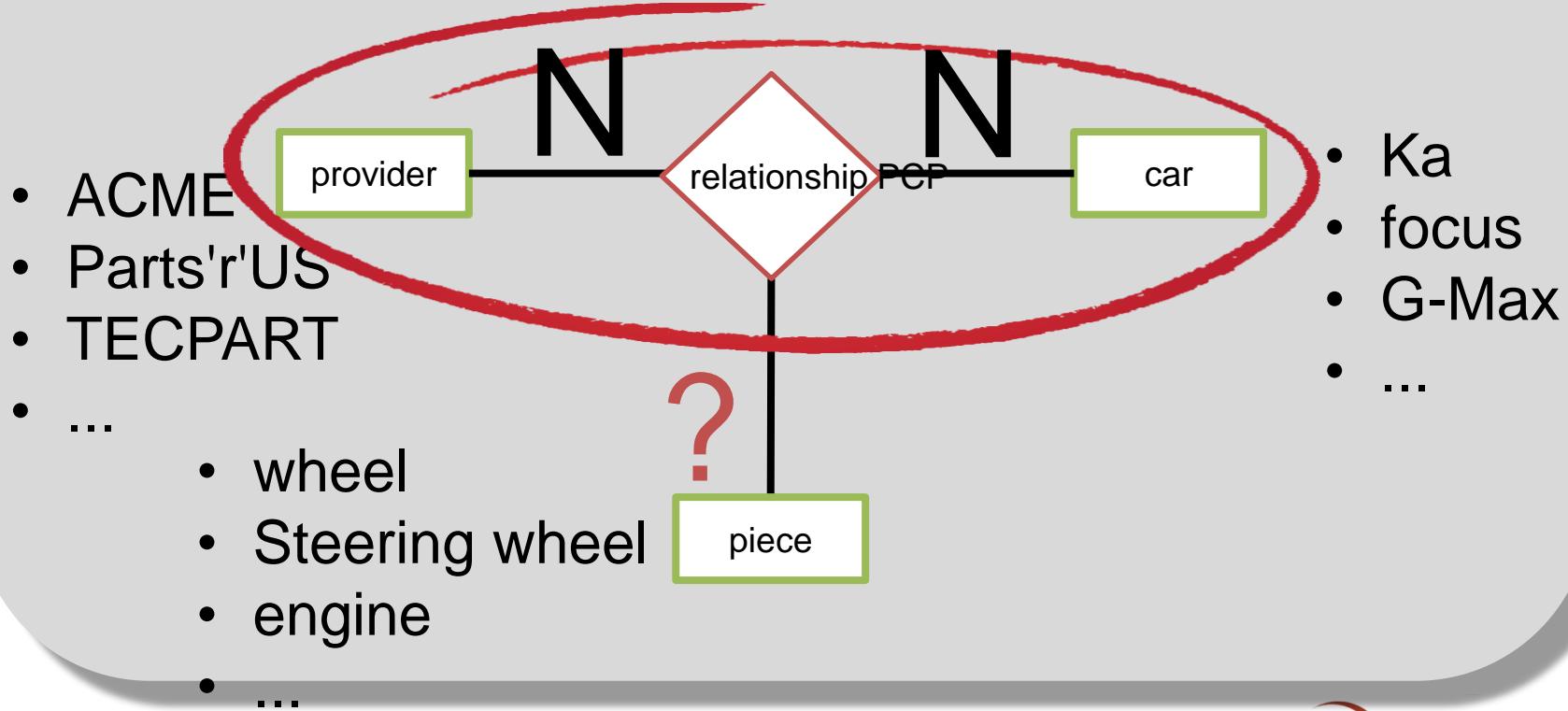
If a combination of car and piece may come from different vendors, then the cardinality of "Provider" is N (many)



# Reviewing Cardinality

## Discussion 2:

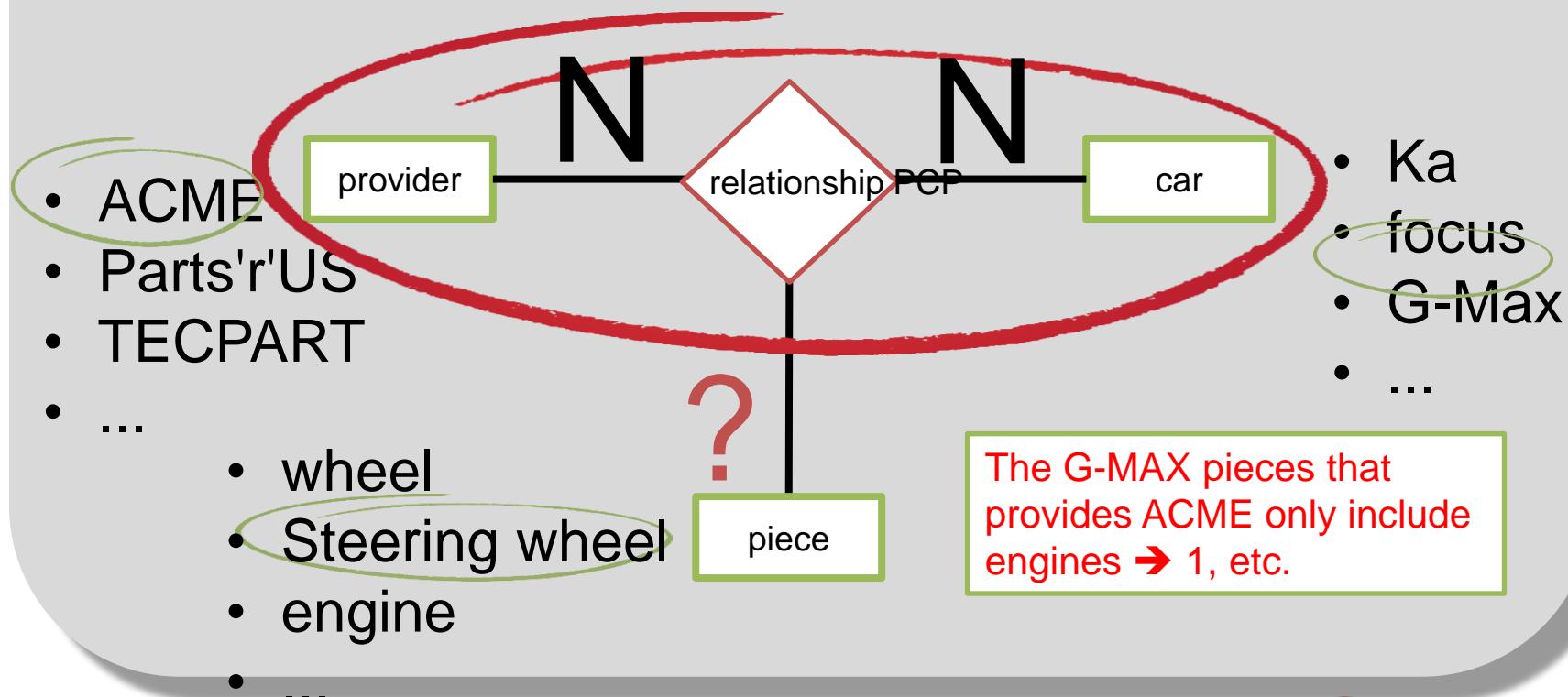
Also for a given instance of a given Supplier and a given Car  
... Can this particular combination (Provider / car) relate to various pieces or at most to just one piece?



# Reviewing Cardinality

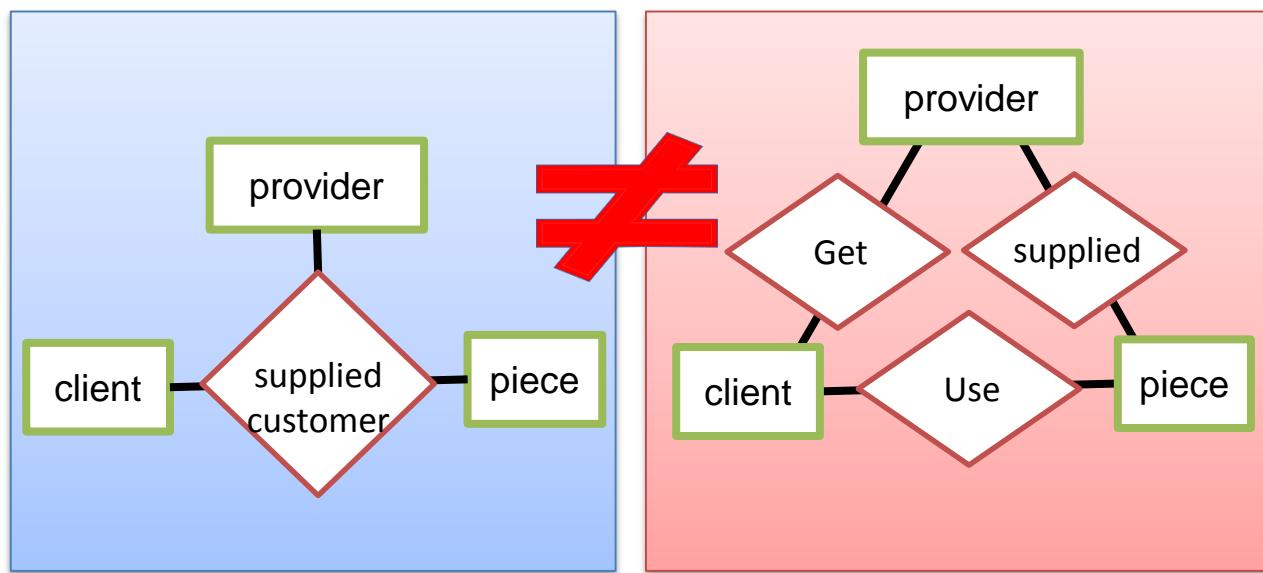
## Discussion 2:

Also for a given instance of a given Supplier and a given Car  
... Can this particular combination (Provider / car) relate to various pieces or at most to just one piece?

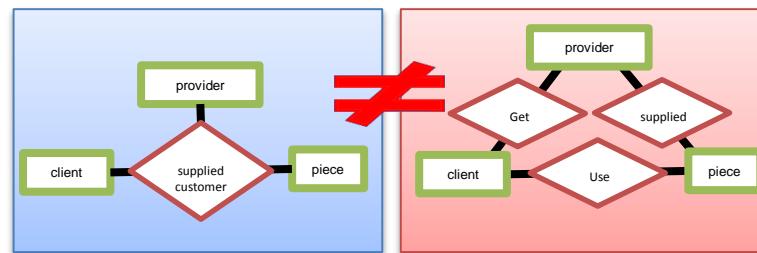


# Conceptual Model: Entity-Relation Design

In this case it is not the same a ternary relation that a three binary relations:



# Conceptual Model: Entity-Relation Design



## Example 1:

The ternary relationship between entities "supplier", "piece" and "client":

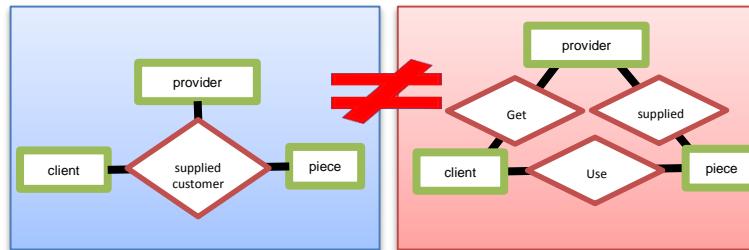
**Some vendors provide certain items to certain customers.**

**NOT semantically equivalent to:**

- Providers SUPPLY Pieces
- Customers use Pieces
- CLIENT receives supplies from provider

} binary relations

# Conceptual Model: Entity-Relation Design



Consider the following individual instances:

- Provider= IKEA,
- Pieces= hinges,
- Client= UAB

IKEA supplies hinges  
to UAB

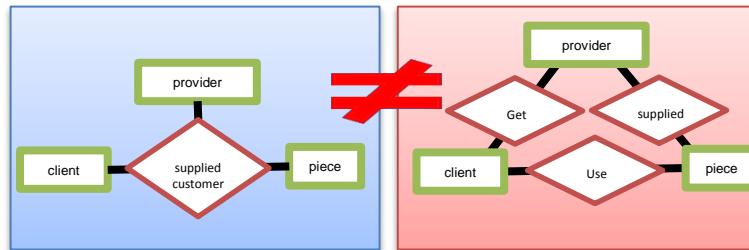
It is semantically  
Equivalent to:

?

binary  
relations

- IKEA supplies hinges
- Hinges are used for UAB
- UAB receives IKEA products

# Conceptual Model: Entity-Relation Design



Consider the following individual instances:

- Provider= IKEA,
- Pieces= hinges,
- Client= UAB

IKEA supplies hinges  
to UAB

It is semantically  
Equivalent to:

?

can not be  
deduced

binary  
relations

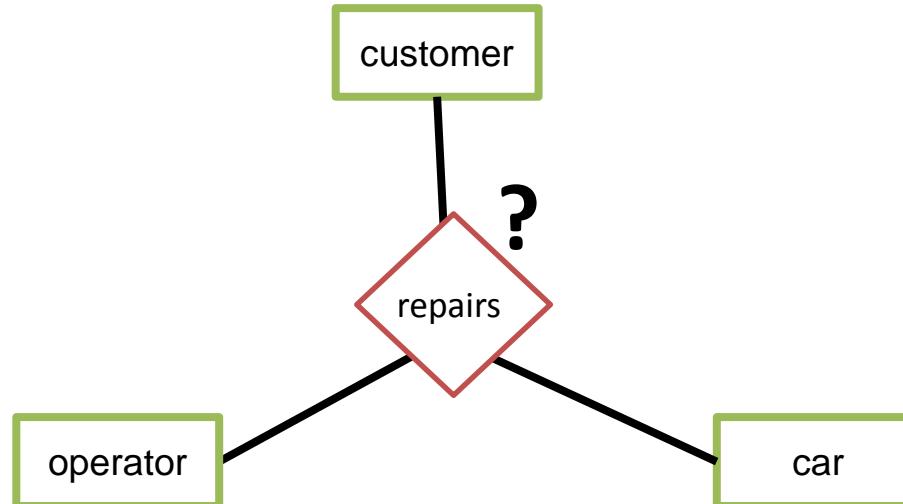
- IKEA supplies hinges
- Hinges are used for UAB
- UAB receives IKEA products

False inference: TRAP CONNECTION

# Conceptual Model: Entity-Relation Design

## Example 2

**Car repair:** A car repair shop wants to keep customer information, the operator that serves them and the car being repaired



# Conceptual Model: Entity-Relation Design

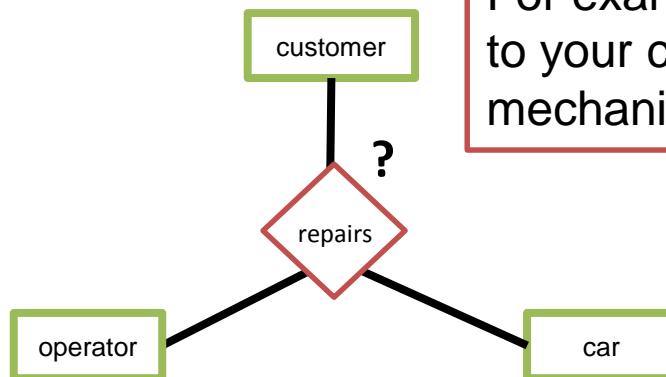
## Example 2

**Car repair:** A car repair shop wants to keep customer information, the operator that serves them and the car being repaired

The interrelationships with degree  $> 2$  are more specific, but more rigid: we have to know all instances when they are created.

For example, when a new client comes to your car you have to assign a mechanic (operator)

**Not practical**

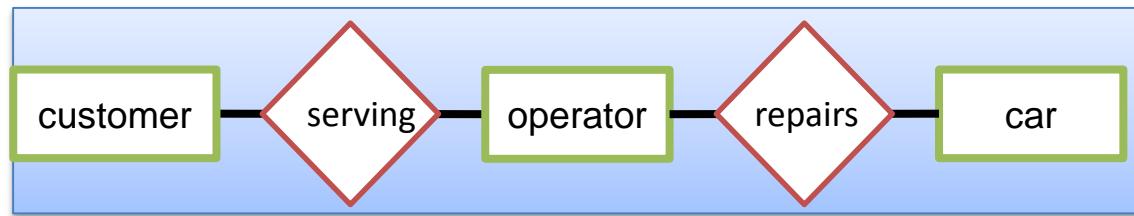


# Conceptual Model: Entity-Relation Design

## Discussion:

**Car repair:** A car repair shop wants to keep customer information, the operator that serves them and the car being repaired

Another possible design is as follows:



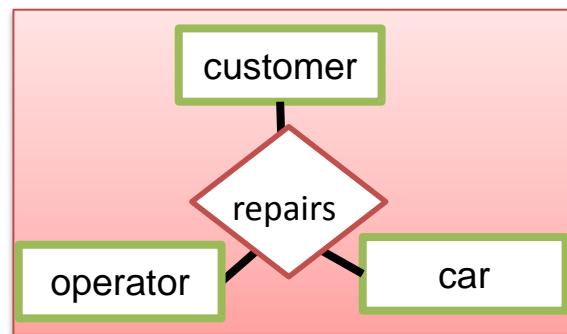
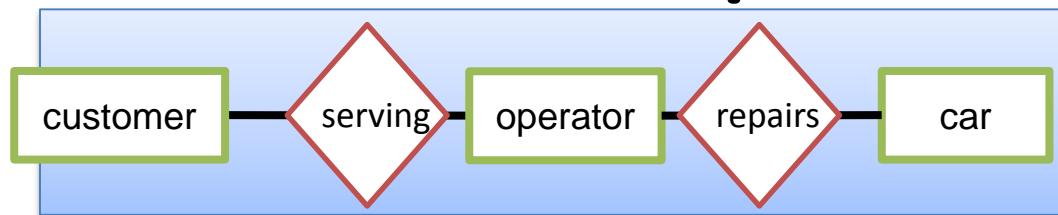
# Conceptual Model: Entity-Relation Design

## Discussion

**Car repair:** A car repair shop wants to keep customer information, the operator that serves them and the car being repaired

These two designs are equivalent

?

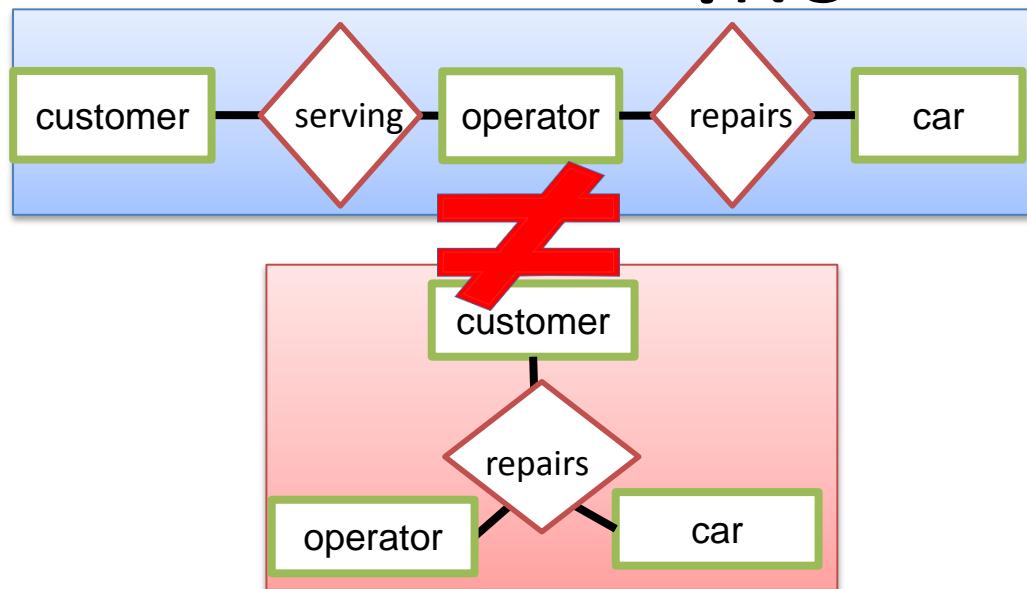


# Conceptual Model: Entity-Relation Design

## Discussion

**Car repair:** A car repair shop wants to keep customer information, and the operator that serves the car being repaired

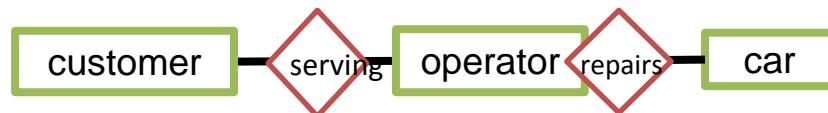
These two designs are equivalent ?**NO**



# Conceptual Model: Entity-Relation Design

## Discussion:

**Car repair:** A car repair shop wants to keep customer information, the operator that serves them and the car being repaired



This scheme allows new customers who do not come for reparation ('partners') **but** I can not relate each car with its owner

I have **ambiguity: Trap connection**

# Conceptual Model: Entity-Relation Design

## Discussion: Trap connection



*example:*

- *Operator 7 serves Mr. Puig*
- *The operator 7 repairs a car with licenses plate 567 HYT*

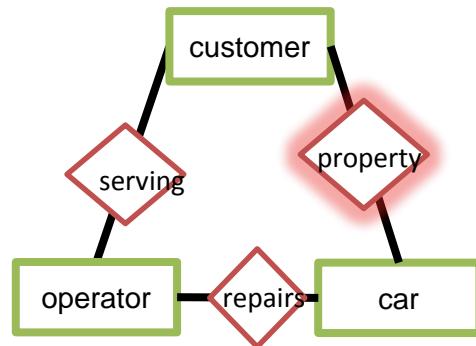
***Is repairing the Mr. Puig car? We don't know. I can not connect cars with their owners (Trap connection)***

# Conceptual Model: Entity-Relation Design

## Discussion: Trap connection



## Solution:



- *Mr. Puig is the owner of license plate HYT 567 car*
- *Operator 7 serves Mr. Puig*
- *The operator 7 repairs a car with license plate HYT 567*

# Basic ER Design Contents

1. E-R Model Introduction

2. Basic Structures

    2.1 Entities

    2.2 Attributes

    2.3 Relationships

3. Relationships Features

    3.1 Cardinality

    3.2 Degree

    3.3 Participation

# Conceptual Model: Entity-Relationship Design

## Participation Restrictions

Specify whether the existence of an entity **A** depends on its relation to another **entity B**.

There are two types:

- **Total participation:** All elements of **A** must be related to some element of **B**.

Indicated by the double line



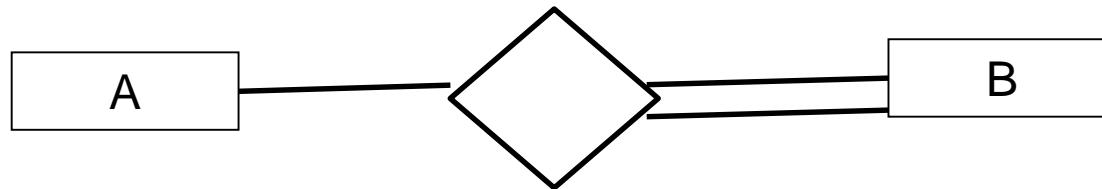
- **Partial participation:** Not all elements of **A** must be related to some element of **B**

Suitable for plain line



# Model Conceptual: Disseny Entitat-Relació (E-R)

Completeness Definition: All B items have be related with some A item. It is represented by a double line from B to the relationship



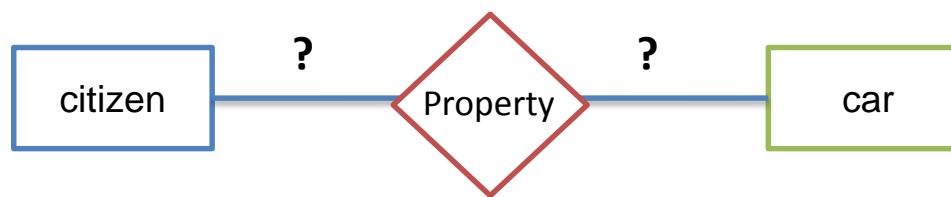
To insert an instance B, I must have inserted an A instance to relate with

# Conceptual Model: Entity-Relation Design

## Participation Restrictions

### Example 1: Relation Car - Citizen

All cars must have an owner



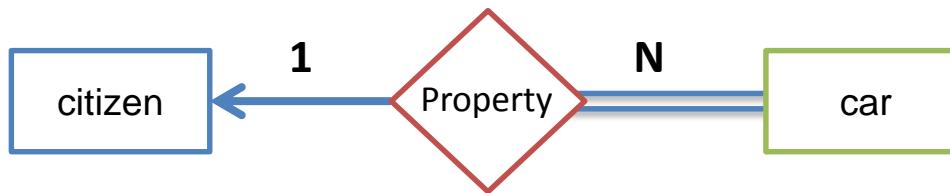
If all cars must have at least one owner (citizen), “car” participation in the relation is total

# Conceptual Model: Entity-Relation Design

## Participation Restrictions

### Example 1: Relation Car - Citizen

All cars must have an owner



Not all citizen have to be car owner. The “citizen” participation in the relation is partial

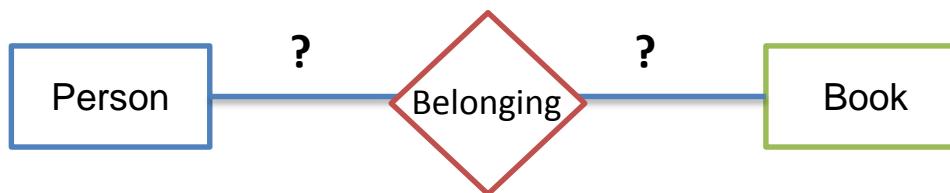
# Conceptual Model: Entity-Relation Design

## Participation Restrictions

The combination cardinality / participation appropriate for a particular relation obviously depends on the situation of the real world of this relationship

### Example 2: Relation Person - Book

Each book should belong to anyone?



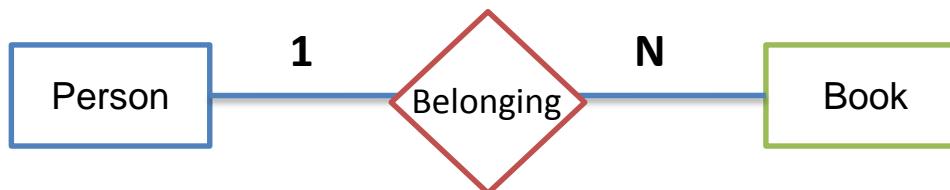
# Conceptual Model: Entity-Relationship Design

## Participation Restrictions

The combination cardinality / participation appropriate for a particular relation obviously depends on the situation of the real world of this relationship

**Example 2:** Relation Person - Book

**Each book should belong to anyone?: NO**



Depends on the context, but in most cases it is not necessary that every book has an owner ...

# Conceptual Model: Entity-Relation Design

## Participation Restrictions

### Example 2: Relation Person - Book

Each book should belong to anyone? :YES



In the case that any book must necessarily have an owner, then participation is "total"

# Block 4

# ADVANCED ER-DESIGN

## (PART 3)

Debora Gil, Oriol Ramos, Carles Sanchez

# Contents: Advanced Design-ER

## 1. Weak Entities

1.1 Definition and Examples

1.2 Design with Weak Entities

## 2. Aggregations

2.1 Specialization

2.2 Generalization

2.3 Aggregation

# Contents: Advanced Design-ER

## 1. Weak Entities

1.1 Definition and Examples

1.2 Design with Weak Entity

## 2. Aggregations

2.1 Specialization

2.2 Generalization

2.3 Aggregation

# Specialization

## Specialization

### Definition:

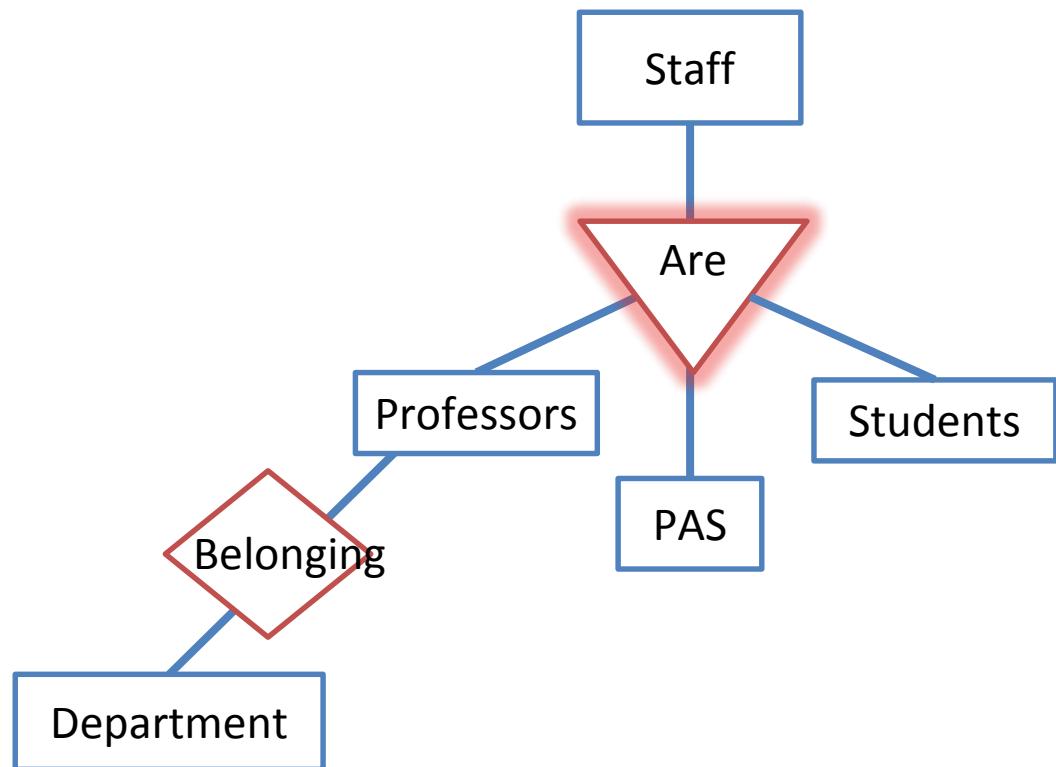
Partition of an entity in specific groups (low-level entities)

An entity may include entities subgroups that are somehow different from other entities. The process of appointment of subgroups within a set of entities is called **Specialization**.

# Specialization

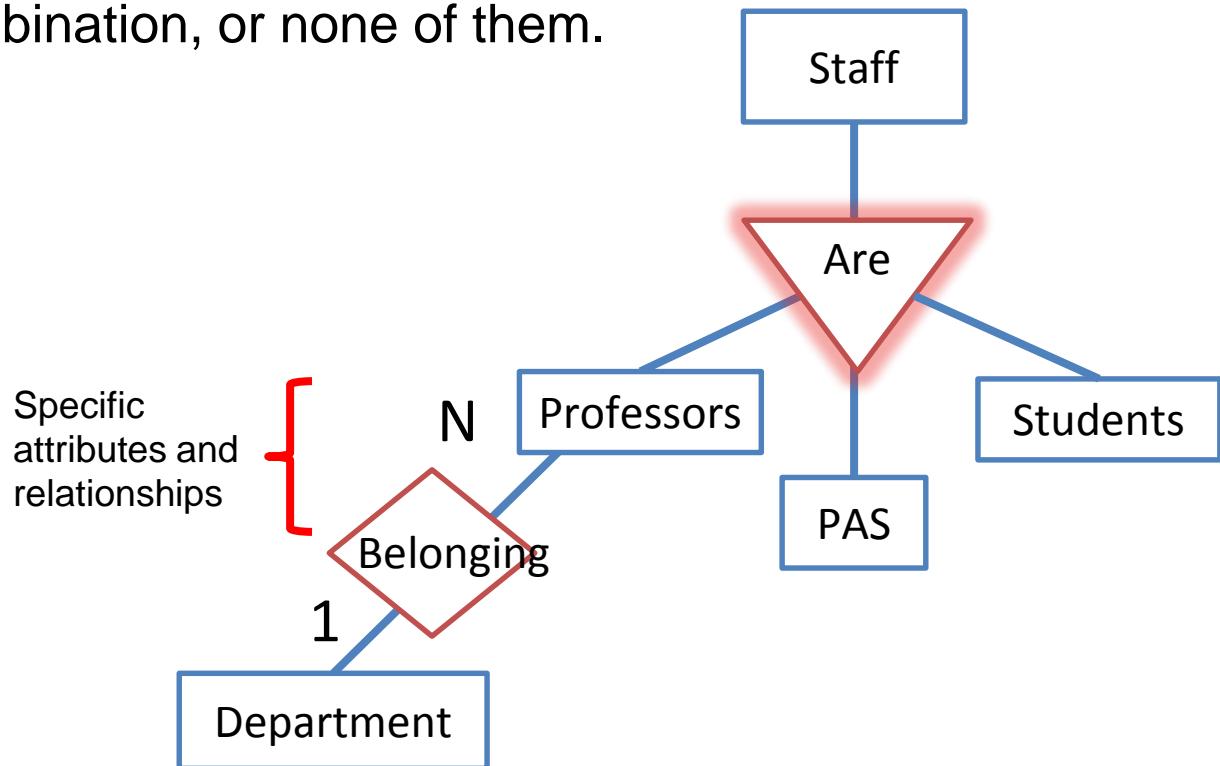
**Example:** instances of “Staff” entity can be classified as one of the following:

- Employees
- Students
- Administration and Services (PAS)



# Specialization

The specialization of "Staff" allows us to distinguish between entities "Staff" according to whether they correspond to the "Teachers", "PAS" or "Students": In general terms, the staff could be an employee, PAS, a student, a combination, or none of them.



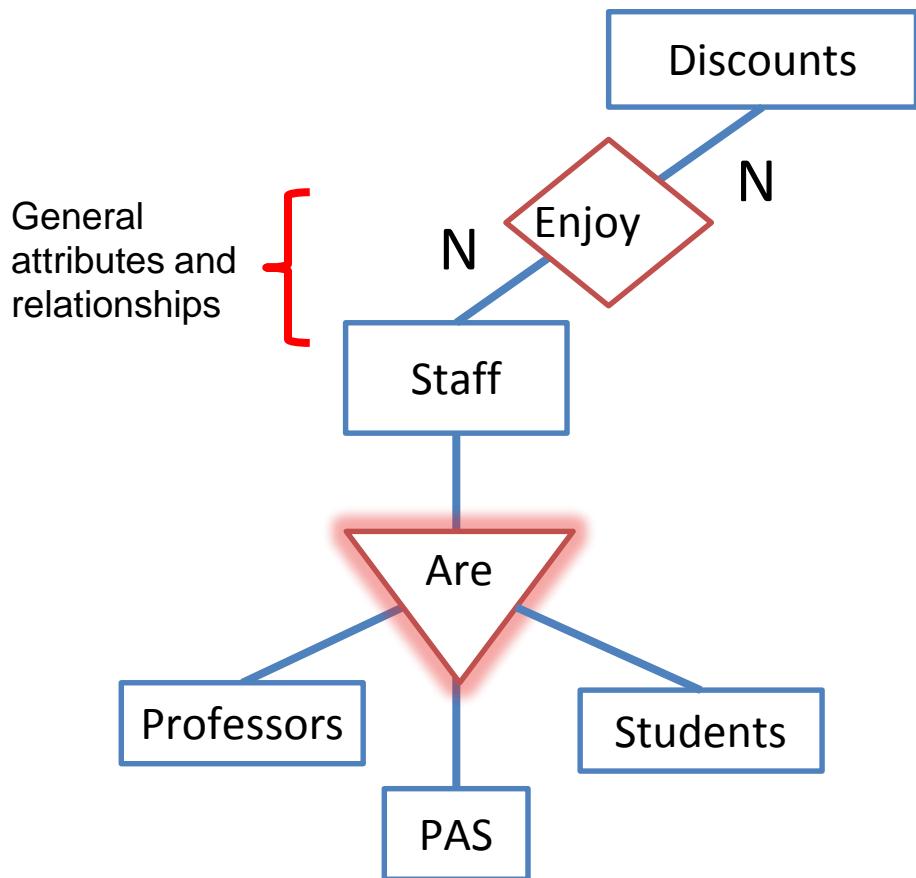
## Generalization

### Definition:

Process summarizing multiple entities into a high level entity based on common characteristics (the reverse process of specialization)

There may be similarities between entity "A" and entity "B", for instance, several attributes that are conceptually the same in both entities. This coincidence can be expressed by **generalization**, which is a containment relationship between an higher level entity and one or multiple lower level entities.

# Generalization



The generalization is used to merge a set of entities that share the same characteristics on a set of higher-level entities.

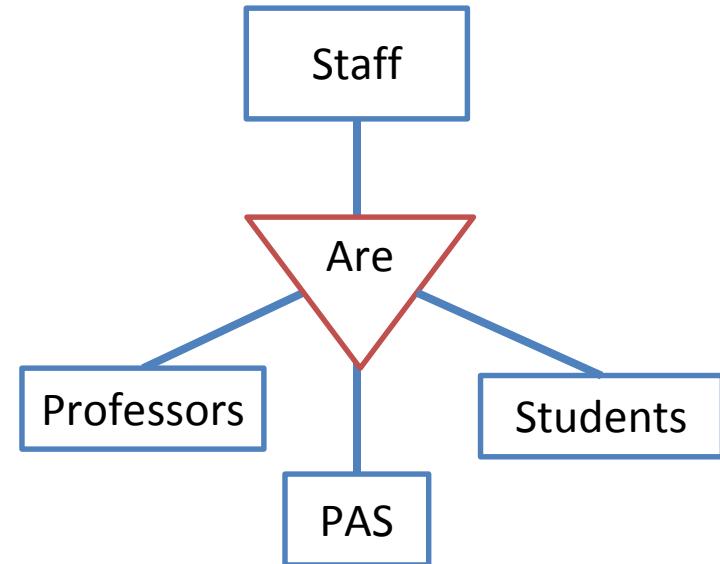
Allows to emphasize the similarities among lower level entities and to hide the differences.

Economic representation: shared attributes are not repeated

# Generalization and Specialization

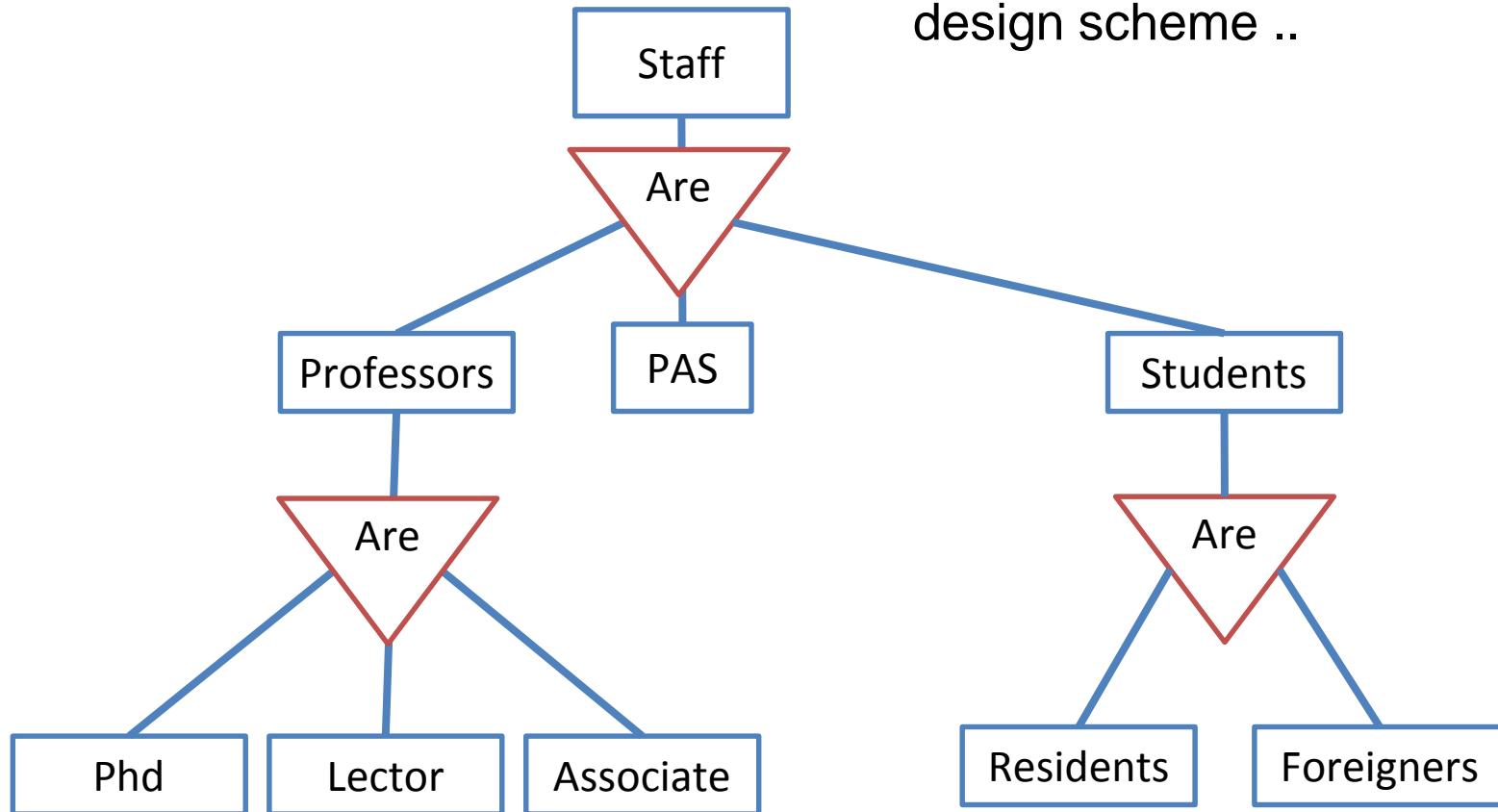
## Common Features

- Grouping of entities with common attributes into a higher level entity
- Eliminate redundancies
- The ER diagram itself does not distinguish between specialization and generalization.
- Attributes of the higher level entities **are inherited** by lower level entities.
- For example, students, teachers and staff inherit “Staff” attributes.



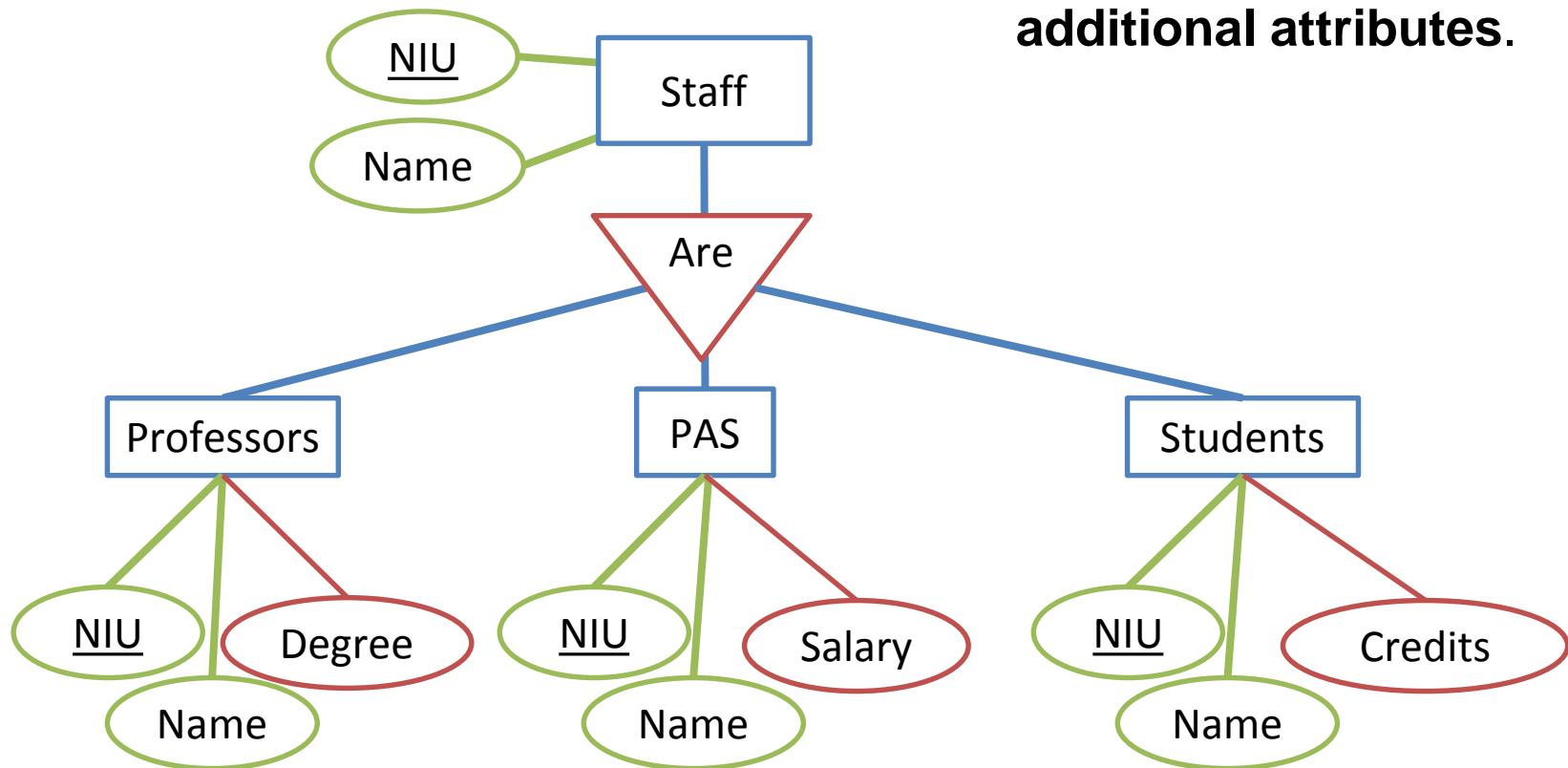
# Generalization and Specialization

We can apply the specialization / generalization more than once to refine a design scheme ..



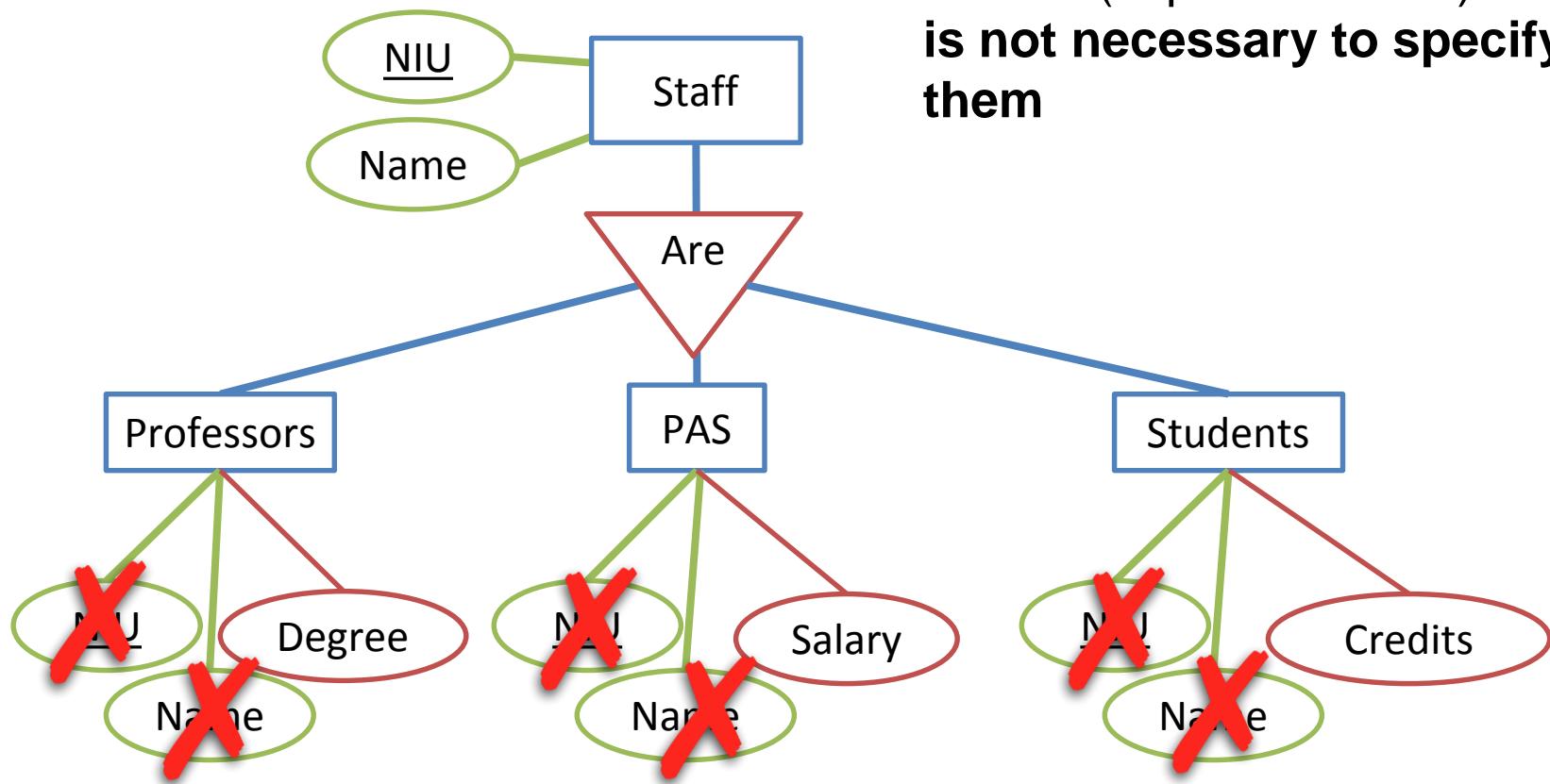
# Generalization and Specialization

Each of these “Staff” sub-entity is described by a set of attributes that includes all the “Staff” attributes adding possibly **some additional attributes**.



# Generalization and Specialization

Attributes and relationships of higher level entities are inherited by the low level entities (in particular CP) and **it is not necessary to specify them**



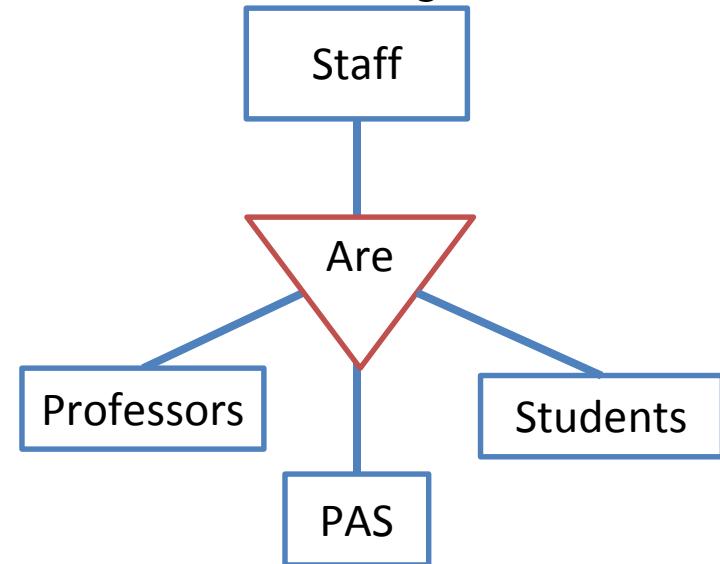
# Generalization and Specialization

## Restrictions

The database designer can decide to place certain restrictions on a particular specialization or generalization

- **Participation:**

- **Total:** All the higher level entity instances must belong to an lower level entity.
- **Partial:** There are instances of the high-level entity that can not be classified (default)



Obs: The generalizations are usually of total participation

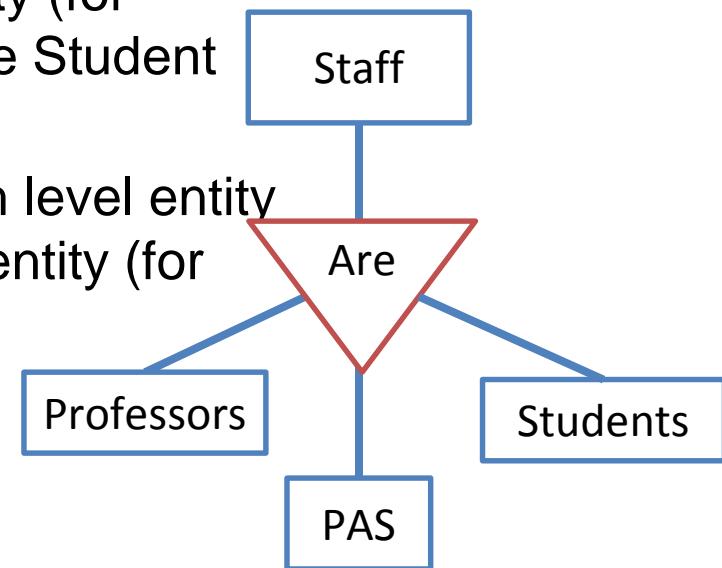
# Generalization and Specialization

## Restrictions

The database designer can decide to place certain restrictions on a particular specialization or generalization

### Partition overlap:

- **Disjointed:** Each instantiation of the high level entity belongs to only one low level entity (for example, a member of “Staff” can not be Student and PAS)
- **Overlapped:** Some instance of the high level entity can belong to more than one low-level entity (for instance, a “Staff” member can be both Student and PAS)



# Generalization and Specialization

## Restrictions

The database designer can decide to place certain restrictions on a particular specialization or generalization

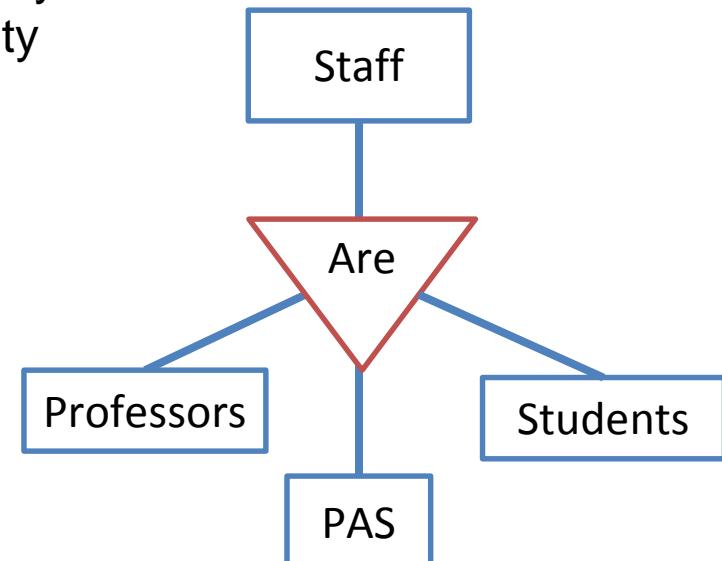
- **Assignment:**

- By **condition**. Some condition is mandatory over some attributes of the high level entity

Example:

- Students: people what are enroled to some course
- Professors: people what teach some course
- PAS: no teaching hired staff

- By **user**: Instance manually assignment



# Aggregation

## Aggregation

### Definition:

Grouping of relationships and participating entities into a new entity

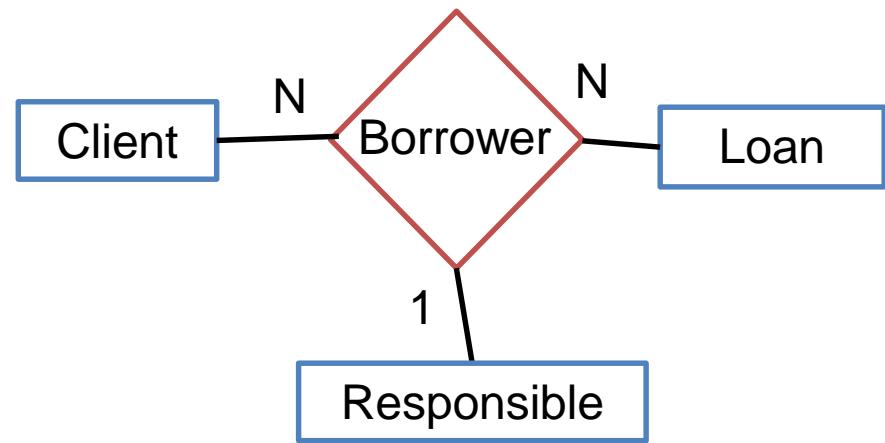
- A limitation of the ER model is that it can not express relationships between relationships
- To solve this problem, we create aggregations that are abstractions through which relationships are treated as higher level entities

# Aggregation

## Example 1: Bank Loan

A bank wants to keep information on loans, their clients and a bank responsible for each loan.

The relationship client-loan is N-N



# Aggregation

## Example 1: Bank Loan

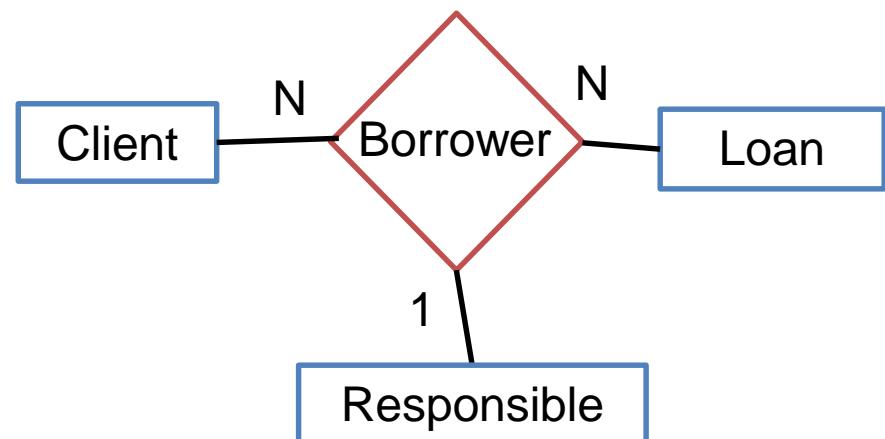
We need to assign the responsible when the loan is accepted

(DNI1, loan1, responsible1)

(DNI1, loan2, responsible1)

(DNI2, loan3, responsible1)

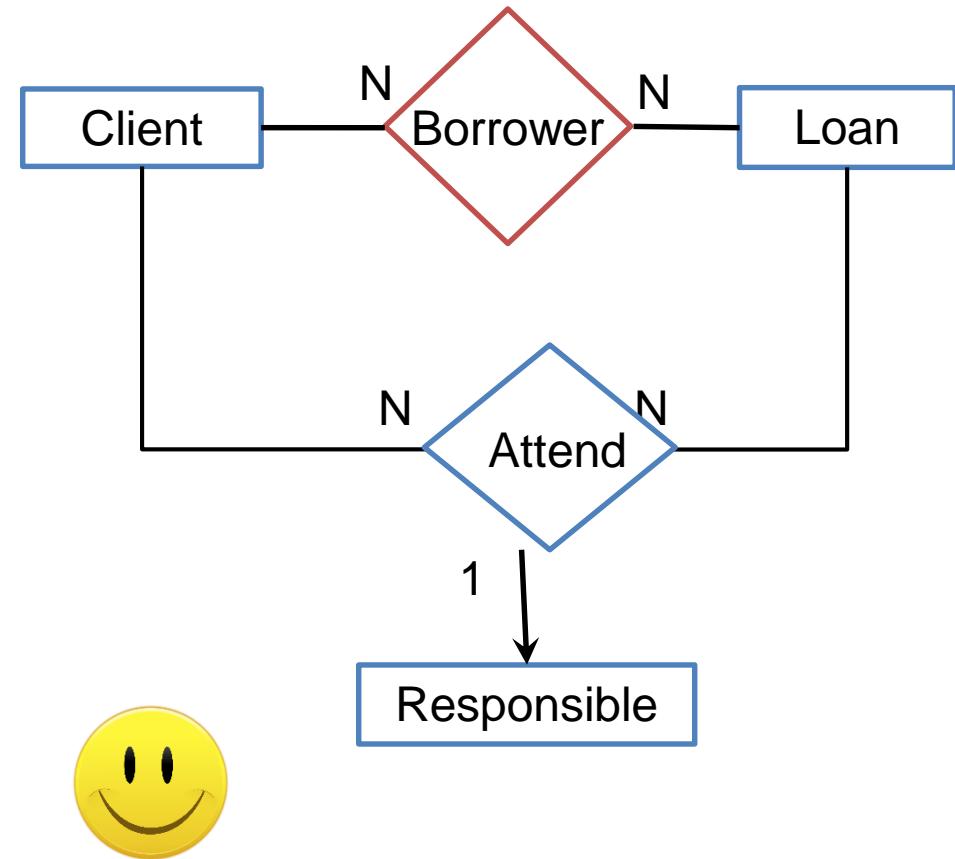
(DNI2, loan4, responsible2)



Very rigid

# Aggregation

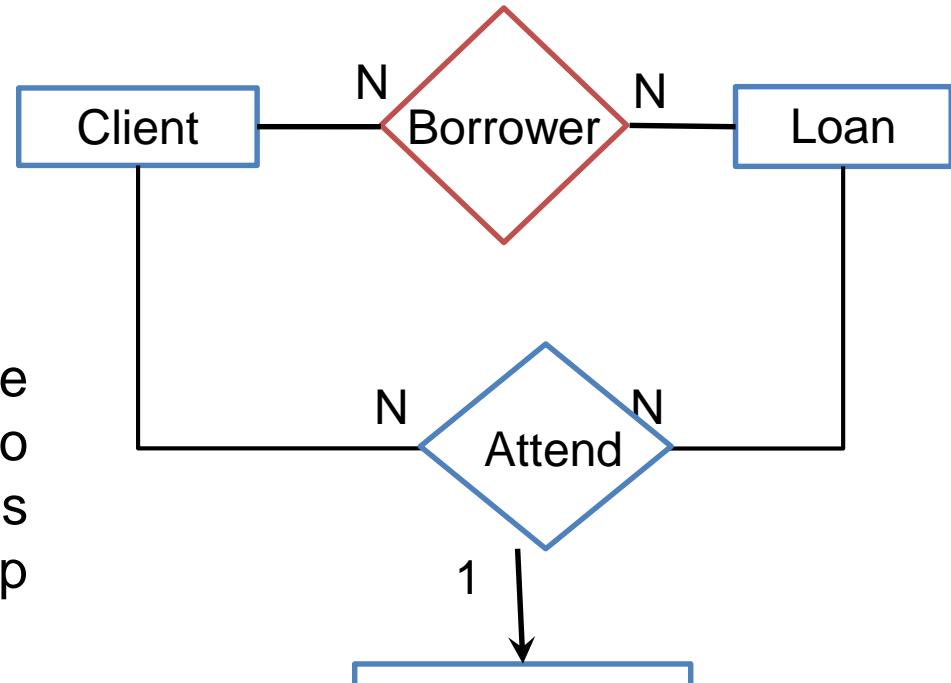
## Example 1: Bank Loan



more convenient

# Aggregation

## Example 1: Bank Loan



**Not as it seems:** we must ensure that every instance of “Attend” is also an instance of “Borrower” and this is not reflected in this diagram (Trap Connection)

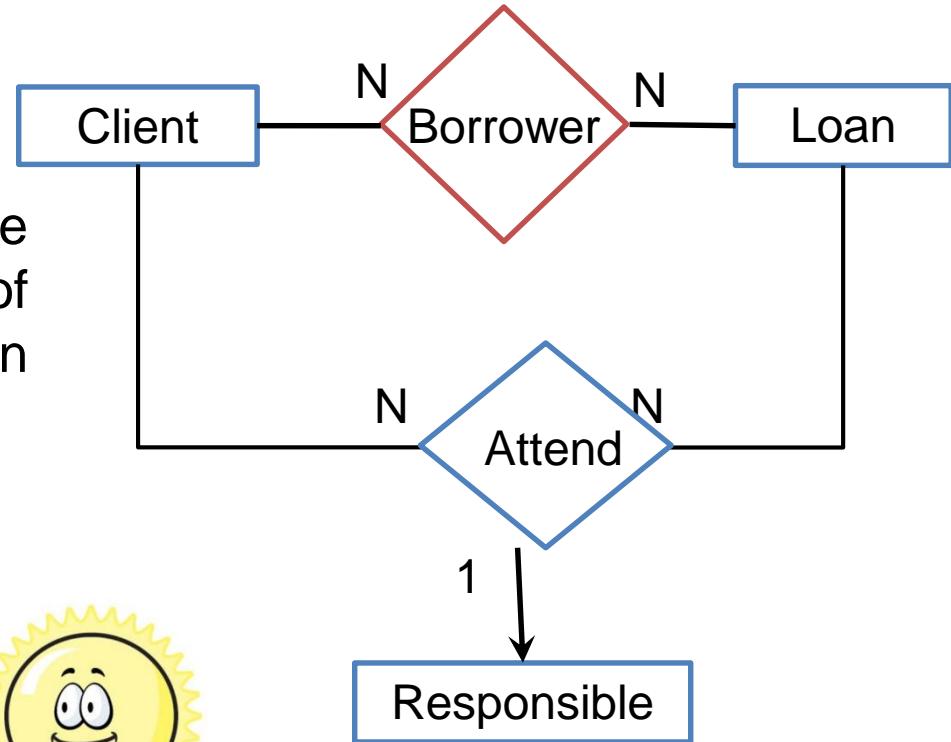


more convenient?

# Aggregation

## Example 1: Bank Loan

We must ensure that every instance of “Attend” is also an instance of “Borrower” and this is not reflected in this diagram (Trap Connection)



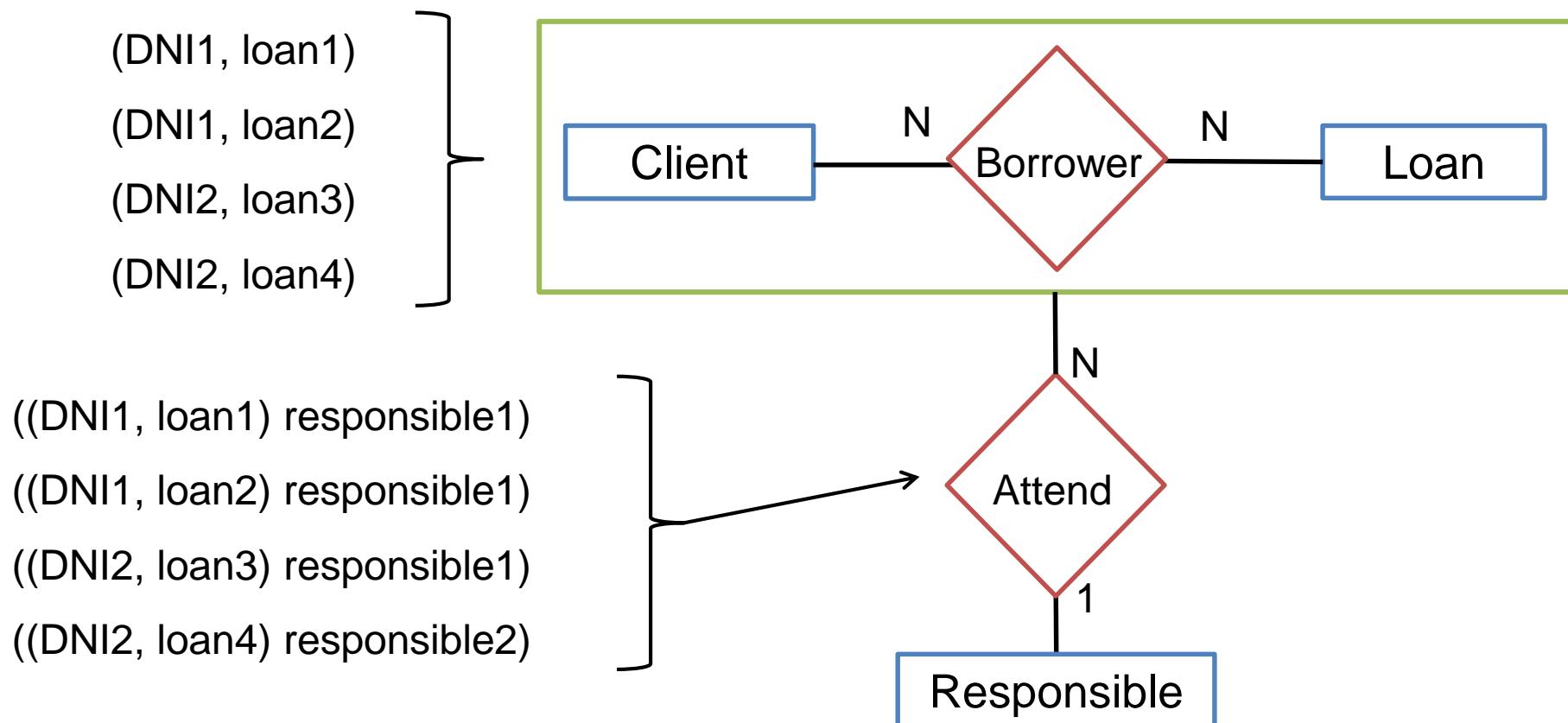
Relationship between  
“Responsible” entity  
and **borrower**  
Relationship.



Solution?

# Aggregation

## Example 1: Bank Loan



# Aggregation

## Example 2: Classrooms booking

We want to manage ETSE classroom timetable booking and know availability. For each school day, reservations are fractions of an hour from 9:00 to 20:00.

For every classroom we keep the code, location and capacity. For each course we want to know the name and degree to which it belongs.

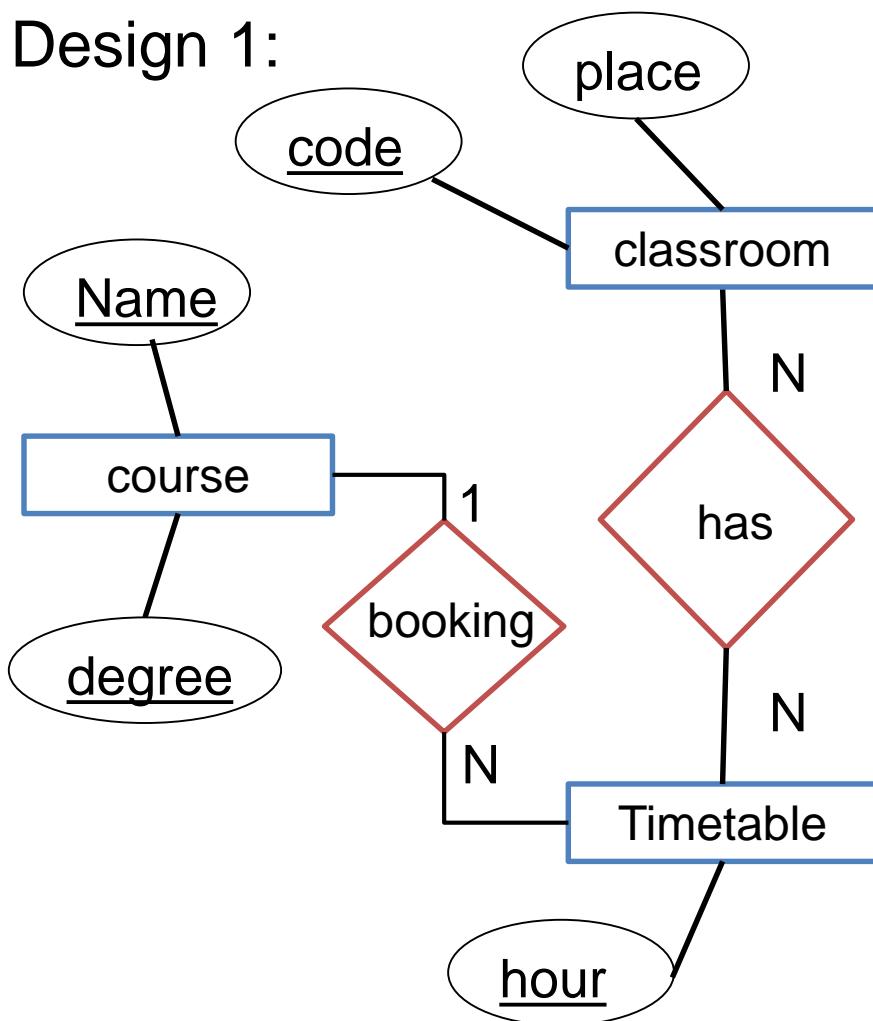
- (a) There are only morning and only afternoon scheduled classrooms
- (b) All classrooms have the same timetable

# Aggregation

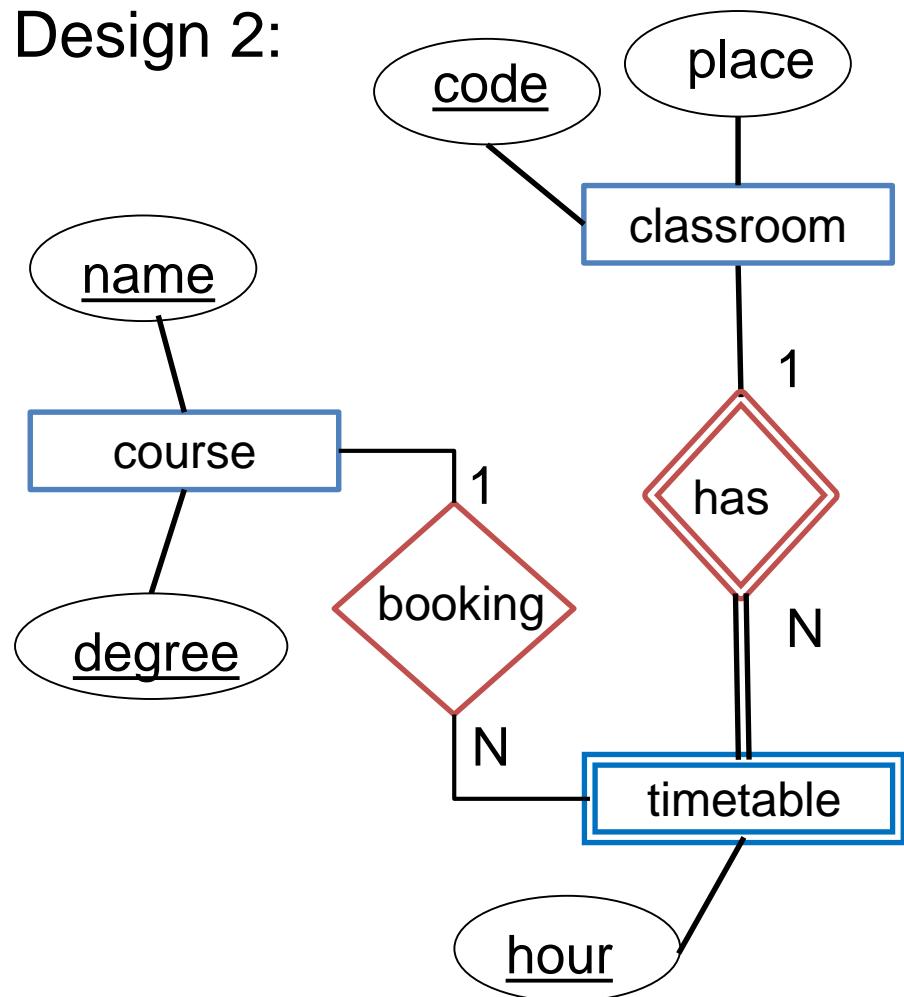
## Example 2: Classrooms booking

We propose two designs:

Design 1:



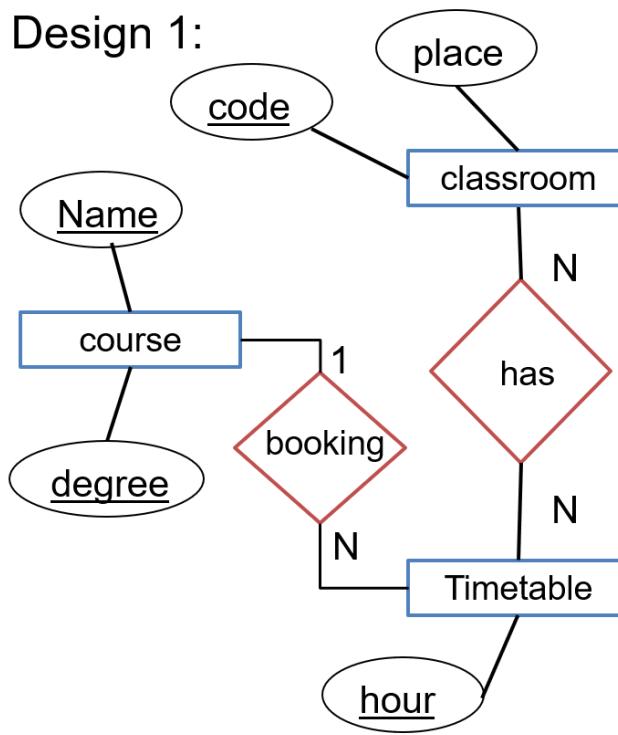
Design 2:



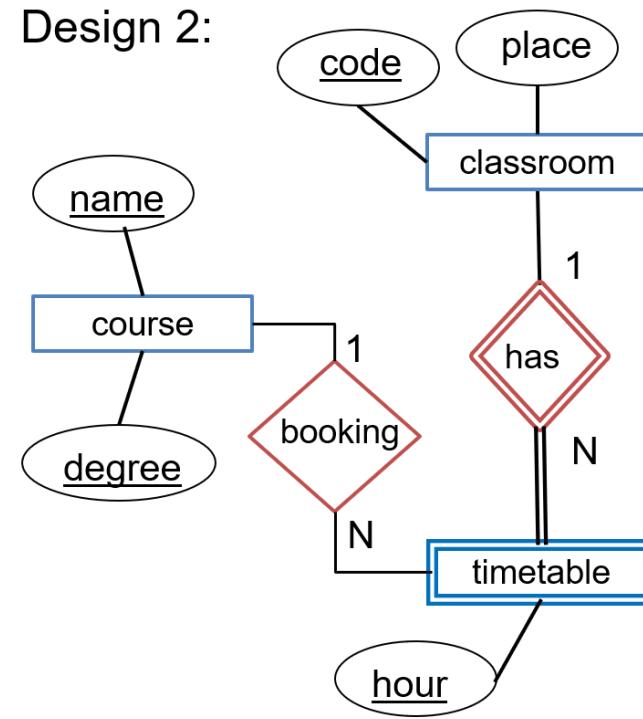
# Aggregation

## Example 2: Classrooms booking

Design 1:



Design 2:



What is the difference between these 2 designs?  
Can we know timetable for requirement (A)? How?

# Aggregation

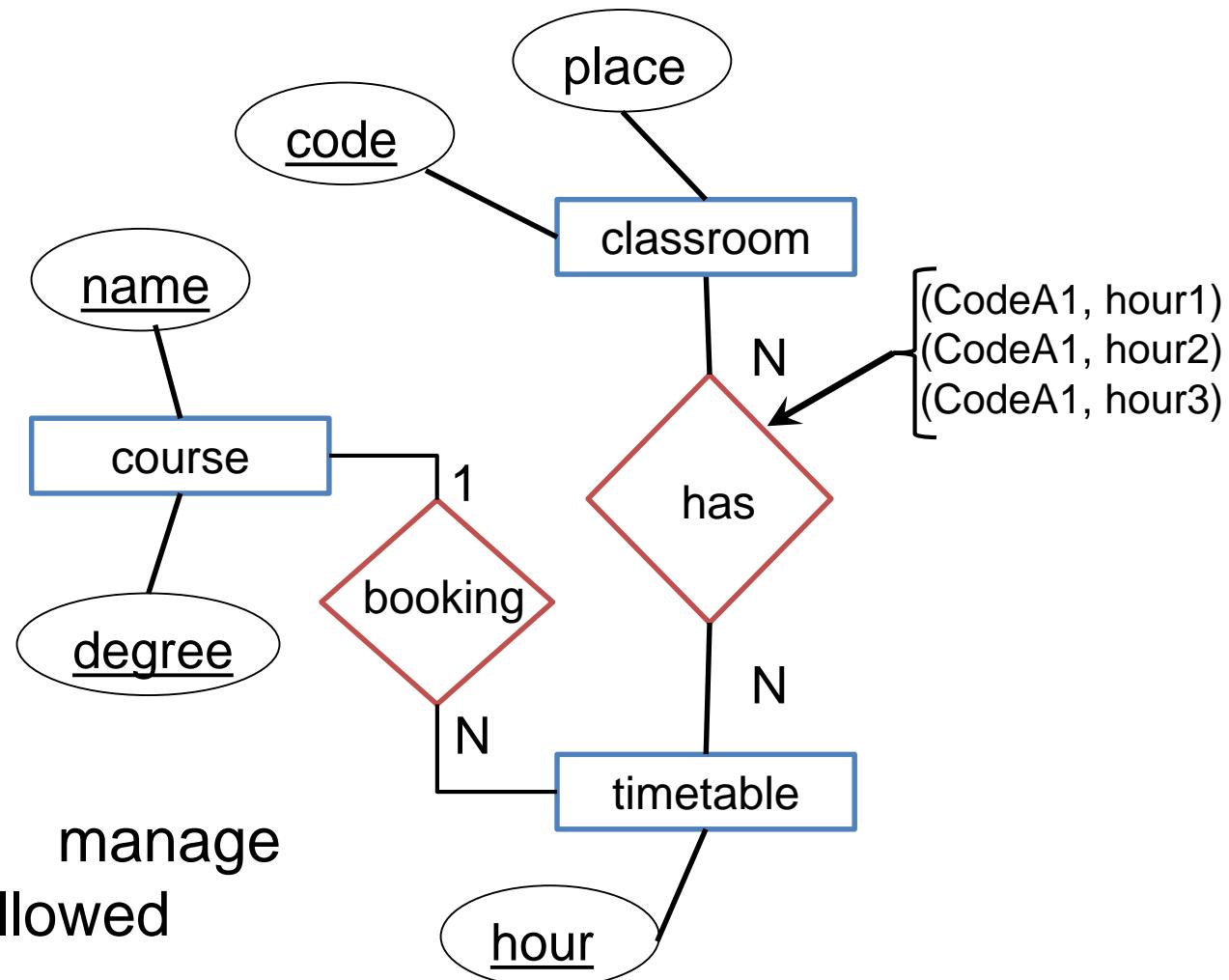
## Example 2: Classrooms booking

If timetable is a strong entity we have generic (list of all slots) timetables, so every classroom timetable ((a) change depending on the classroom) are in the relationship



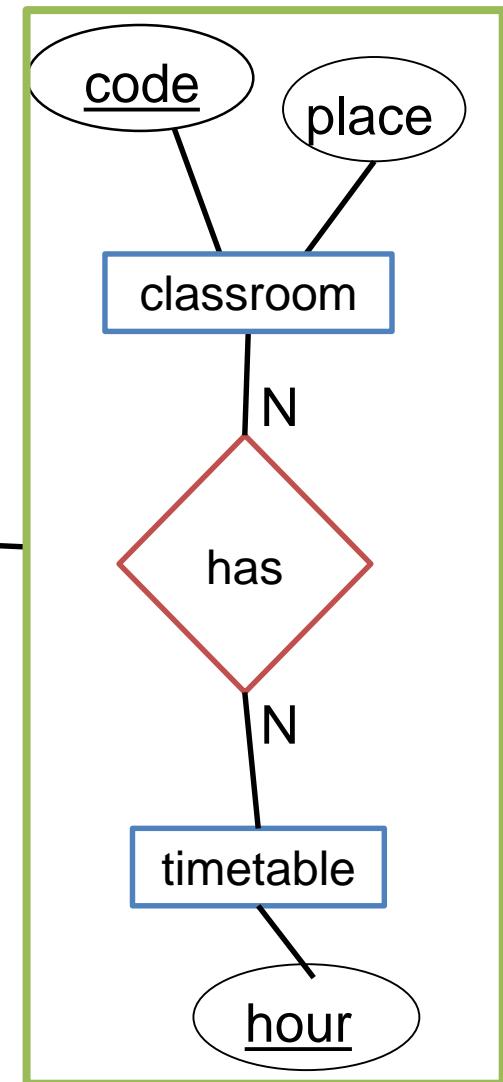
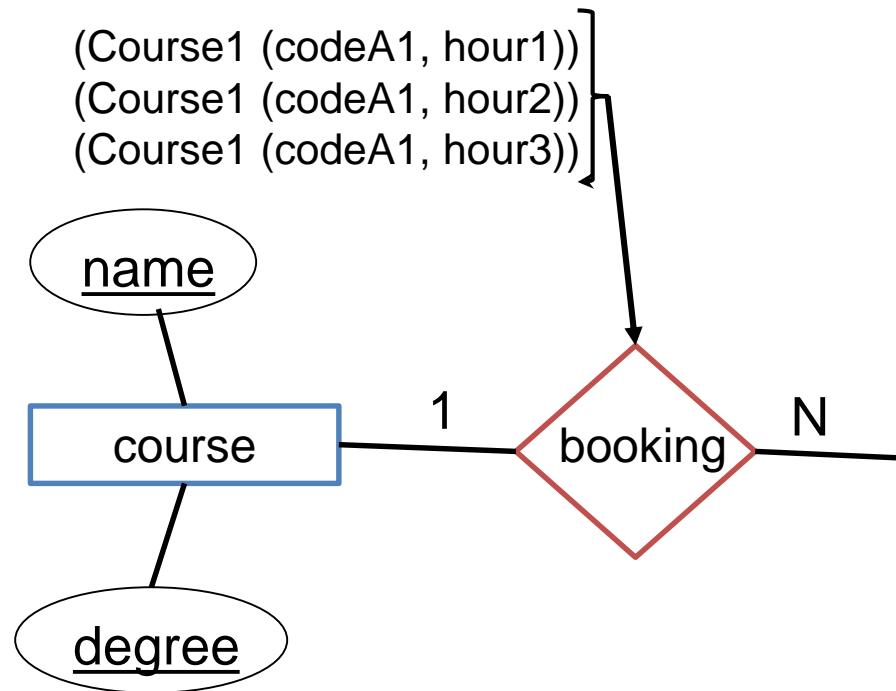
With this design manage availability is not allowed

Design 1:



# Aggregation

## Example 2: Classrooms booking



An aggregation would manage the hours available, although it is more complex

# Aggregation

## Example 3: business operations

A company has a fleet of cars for employees for perform operations outside their workplace. In addition, each time an employee uses a car generates a sheet with all the associated costs.

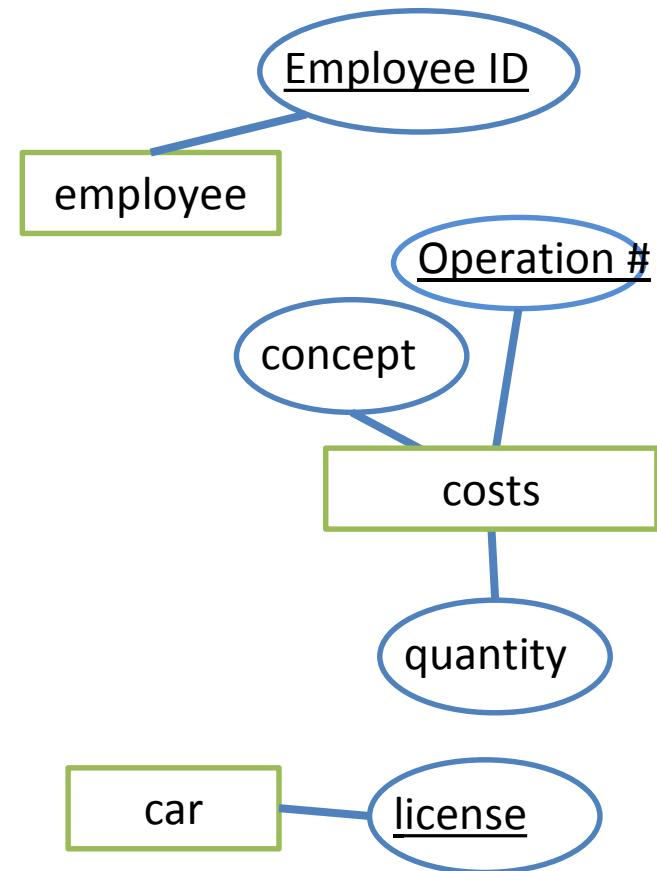
### Particularly, for each operation:

- May be required more than one worker
- No limits on the number of cars in use
- The cost sheet should indicate all associated costs
- One of workers involved in the operations (the manager) is in charge (before use) to reserve cars (among those available), and once the operation has been completed, delivering the sheet with all the costs

# Aggregation

## Example 3: business operations

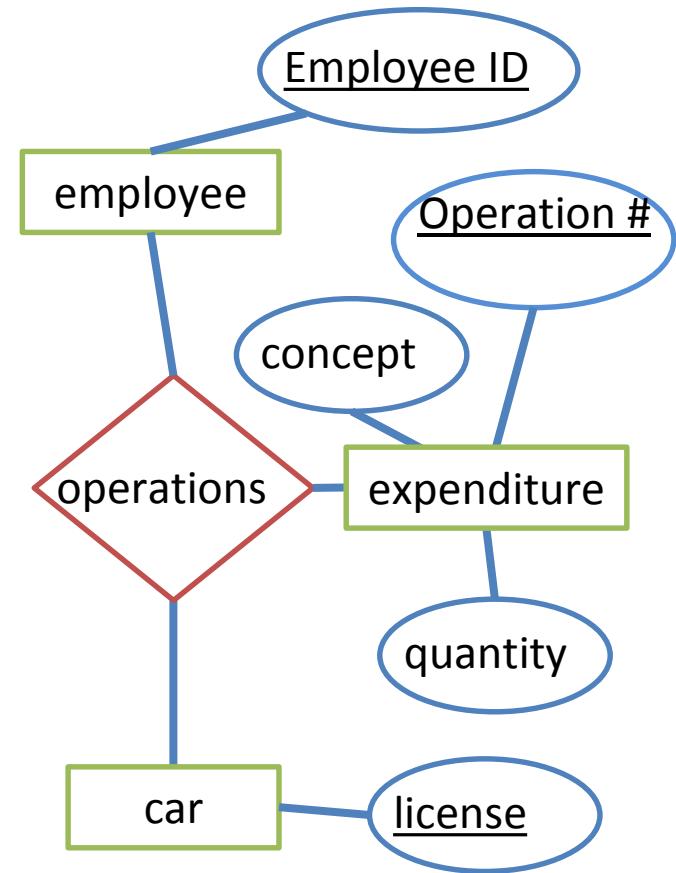
A company has a fleet of **cars** for **employees** for perform operations outside their workplace. In addition, each time an employee uses a car generates a sheet with all the associated **costs**.



# Aggregation

## Example 3: business operations

A company has a fleet of cars for employees for perform **operations** outside their workplace. In addition, each time an employee uses a car generates a sheet with all the associated costs.



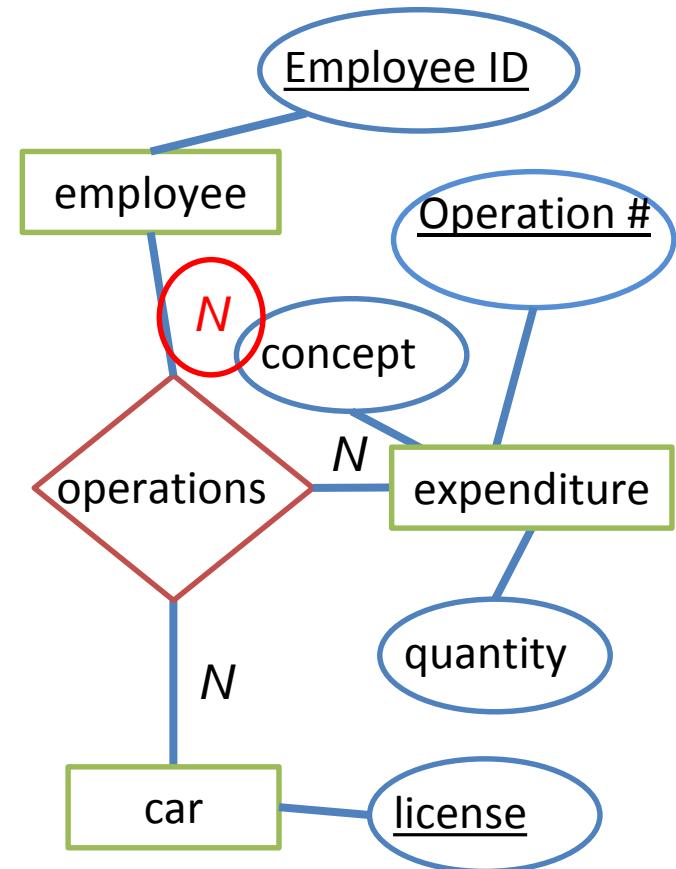
# Aggregation

## Example 3: business operations

A company has a fleet of cars for employees for perform **operations** outside their workplace. In addition, each time an employee uses a car generates a sheet with all the associated costs.

### Particularly, each operation:

- May be required more than one worker
- No limits on the number of cars in use
- The cost sheet should indicate all associated costs
- .....



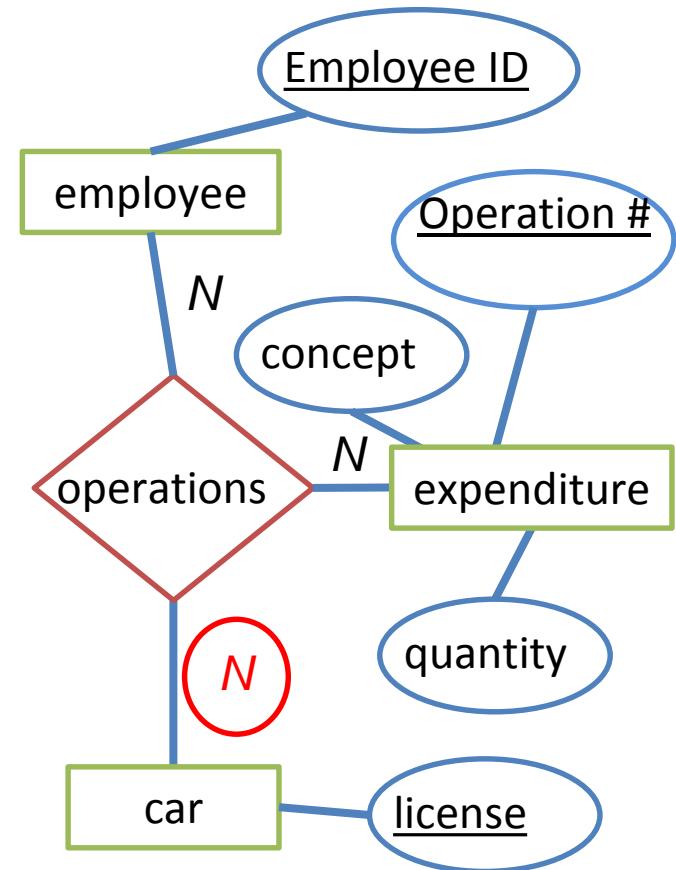
# Aggregation

## Example 3: business operations

A company has a fleet of cars for employees for perform **operations** outside their workplace. In addition, each time an employee uses a car generates a sheet with all the associated costs.

### Particularly, each operation:

- May be required more than one worker
- **No limits on the number of cars in use**
- The cost sheet should indicate all associated costs
- .....



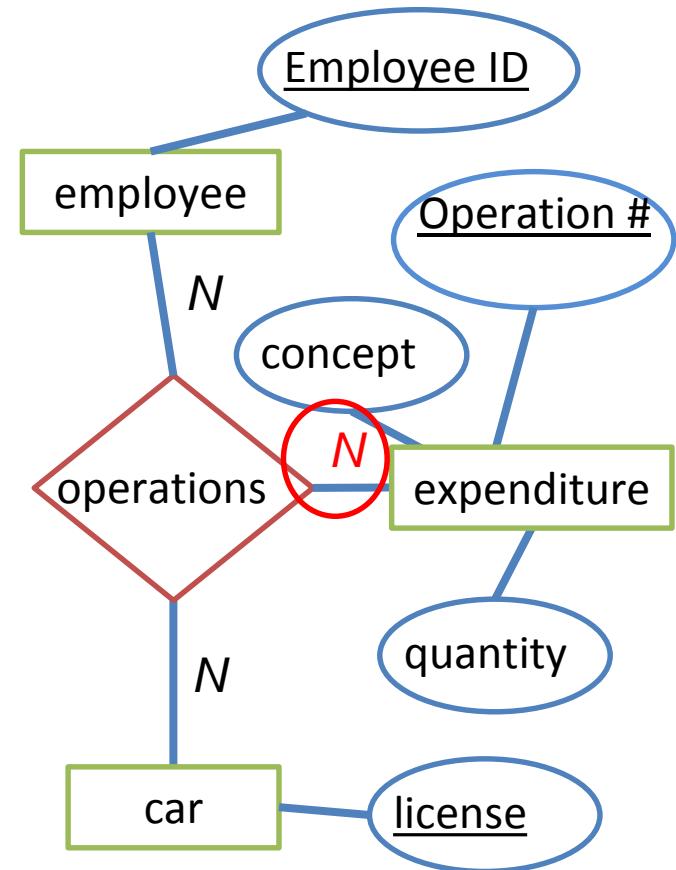
# Aggregation

## Example 3: business operations

A company has a fleet of cars for employees for perform **operations** outside their workplace. In addition, each time an employee uses a car generates a sheet with all the associated costs.

### Particularly, each operation:

- May be required more than one worker
- No limits on the number of cars in use
- The cost sheet should indicate all associated costs
- .....



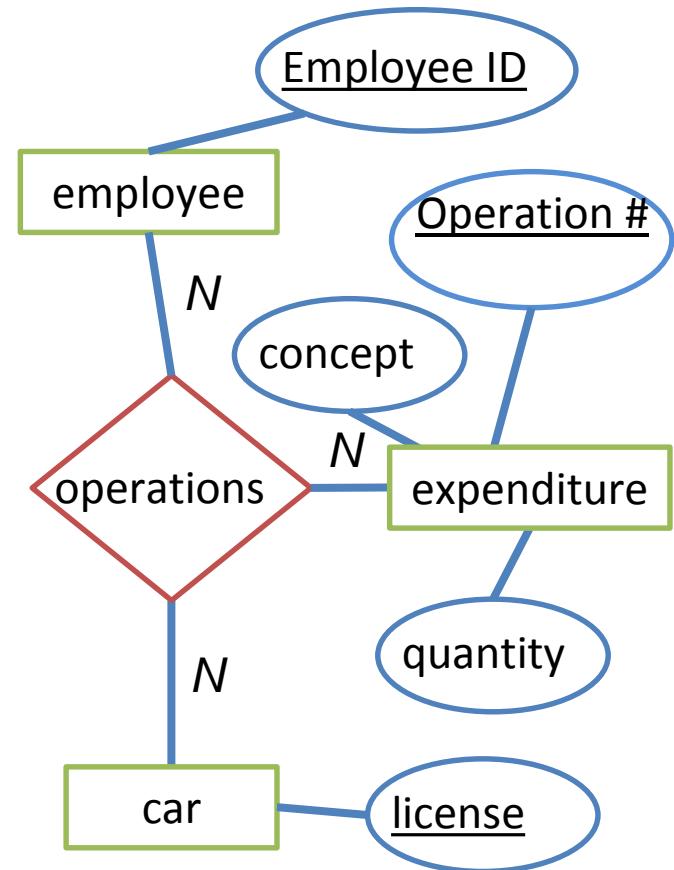
# Aggregation

## Example 3: business operations

A company has a fleet of cars for employees for perform **operations** outside their workplace. In addition, each time an employee uses a car generates a sheet with all the associated costs.

**Particularly, each operation:**

- ...
- One of workers involved in the operations (the manager) is in charge (before use) to reserve cars (among those available), and once the operation has been completed, delivering the sheet with all the costs



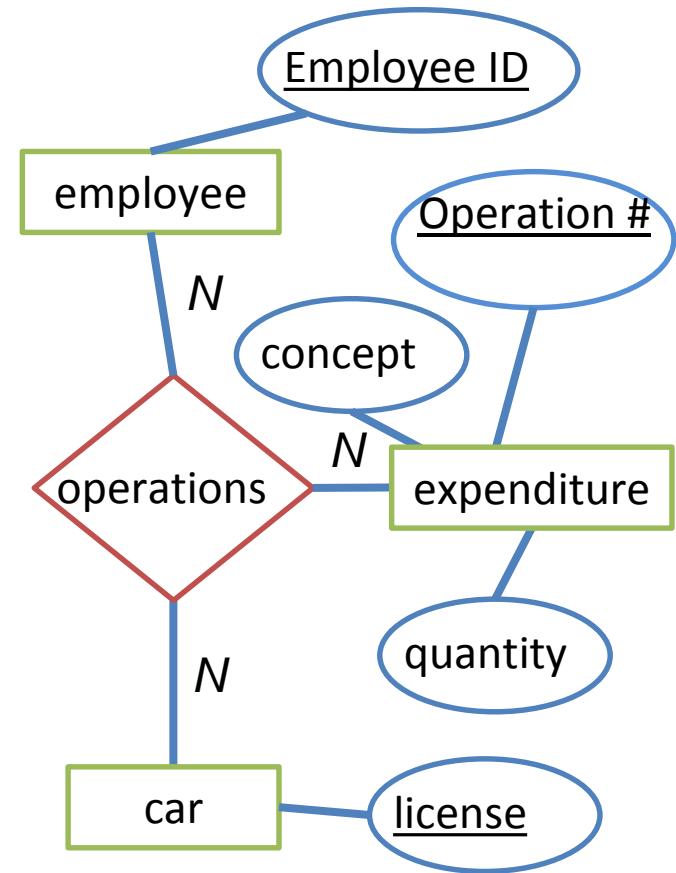
# Aggregation

## Example 3: business operations

A company has a fleet of cars for employees for perform **operations** outside their workplace. In addition, each time an employee uses a car generates a sheet with all the associated costs.

### Problems:

- The ternary relationship does not allow booking first and then **to pass expenses**

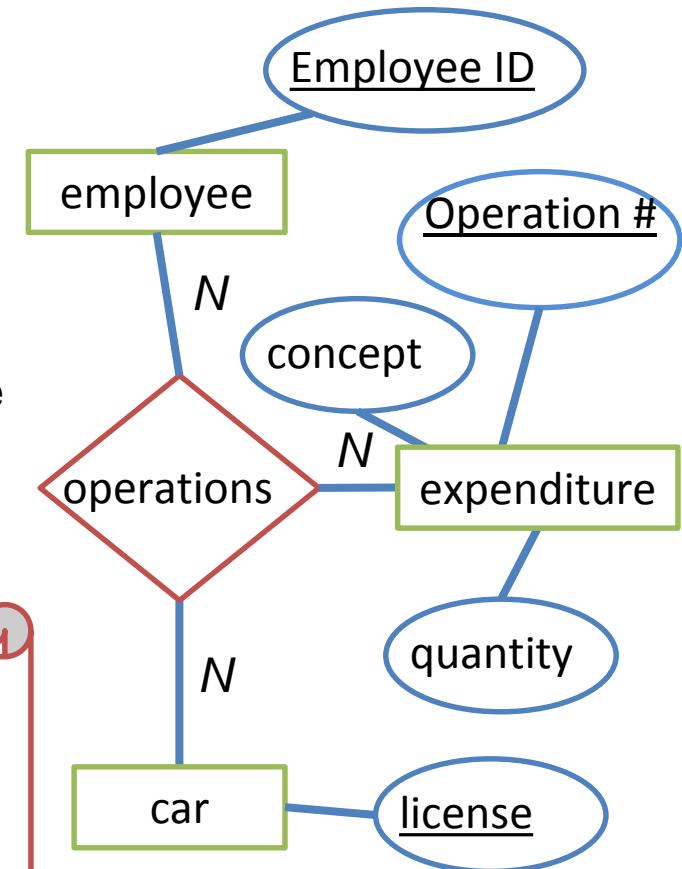


# Aggregation

## Example 3: business operations

A company has a fleet of cars for employees for perform **operations** outside their workplace. In addition, each time an employee uses a car generates a sheet with all the associated costs.

In this case should be allowed operations between **employee** and **car** before issuing costs (requires to be independent "operations" of "spending").



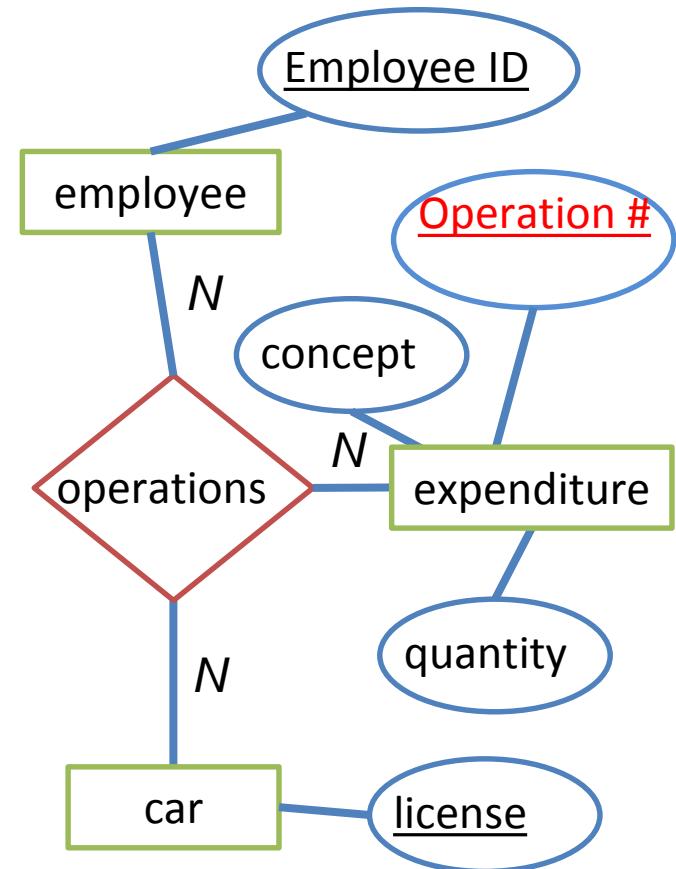
# Aggregation

## Example 3: business operations

A company has a fleet of cars for employees for perform **operations** outside their workplace. In addition, each time an employee uses a car generates a sheet with all the associated costs.

### Problems:

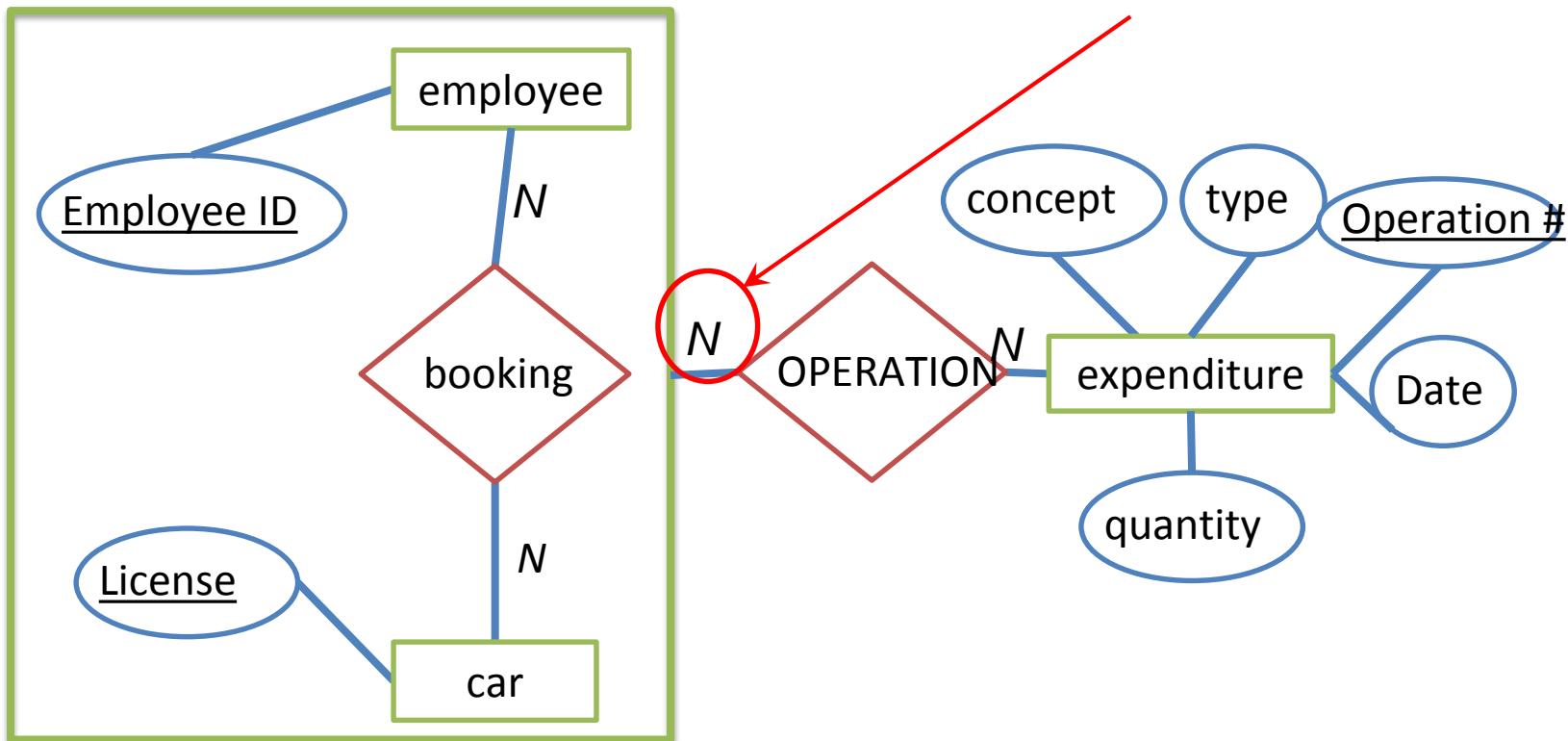
- The ternary relationship does not allow booking first and then **to pass** expenses
- **Costs associated with a real operation** correspond to the same number of operation (**Operation #**) are not checked



# Aggregation

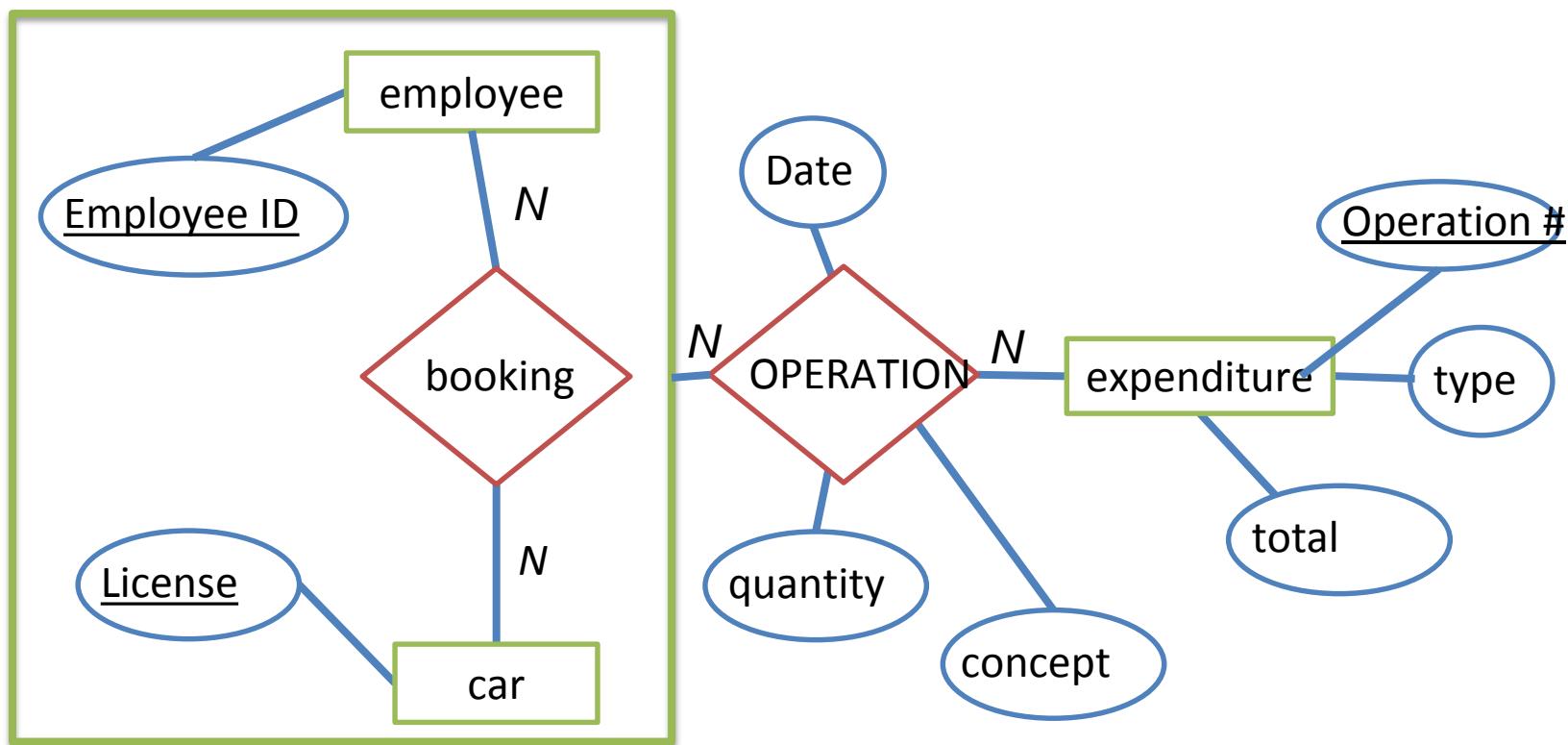
## Example 3: business operations

In one operation, employee can take more than one car



# Aggregation

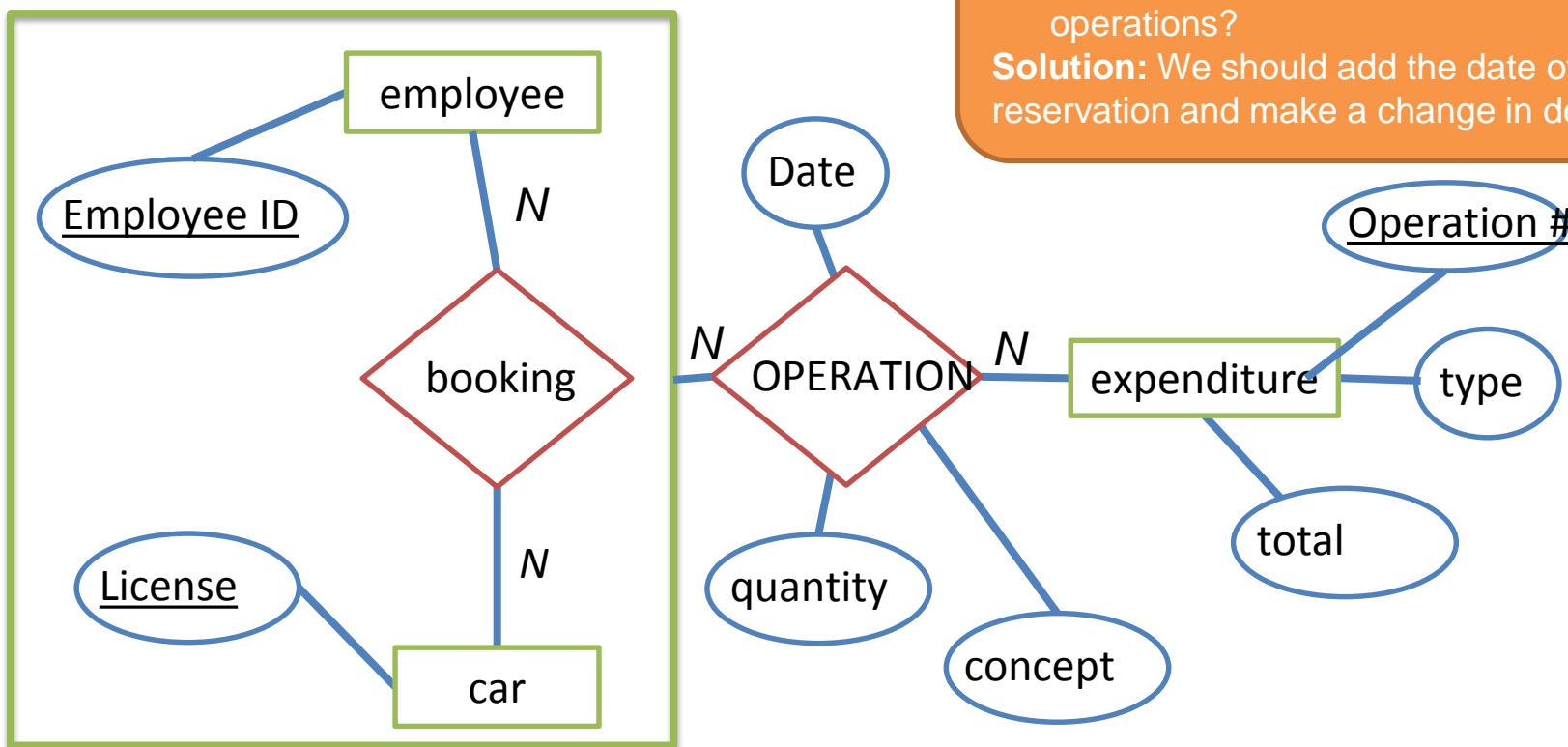
## Example 3: business operations



Databases

# Aggregation

## Example 3: business operations



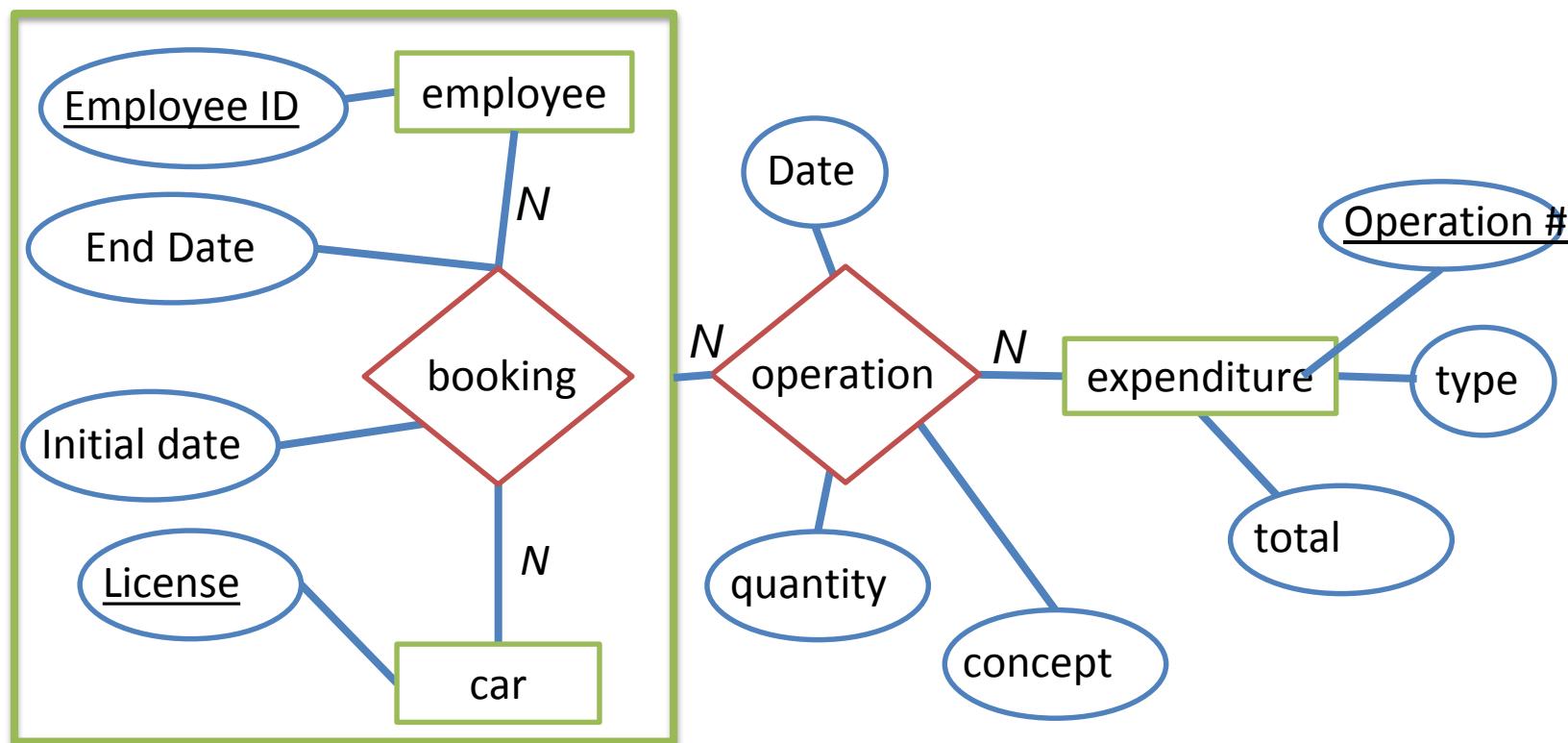
This design still has problems ...

- What happens when an employee takes the same car for different operations?

**Solution:** We should add the date of reservation and make a change in design.

# Aggregation

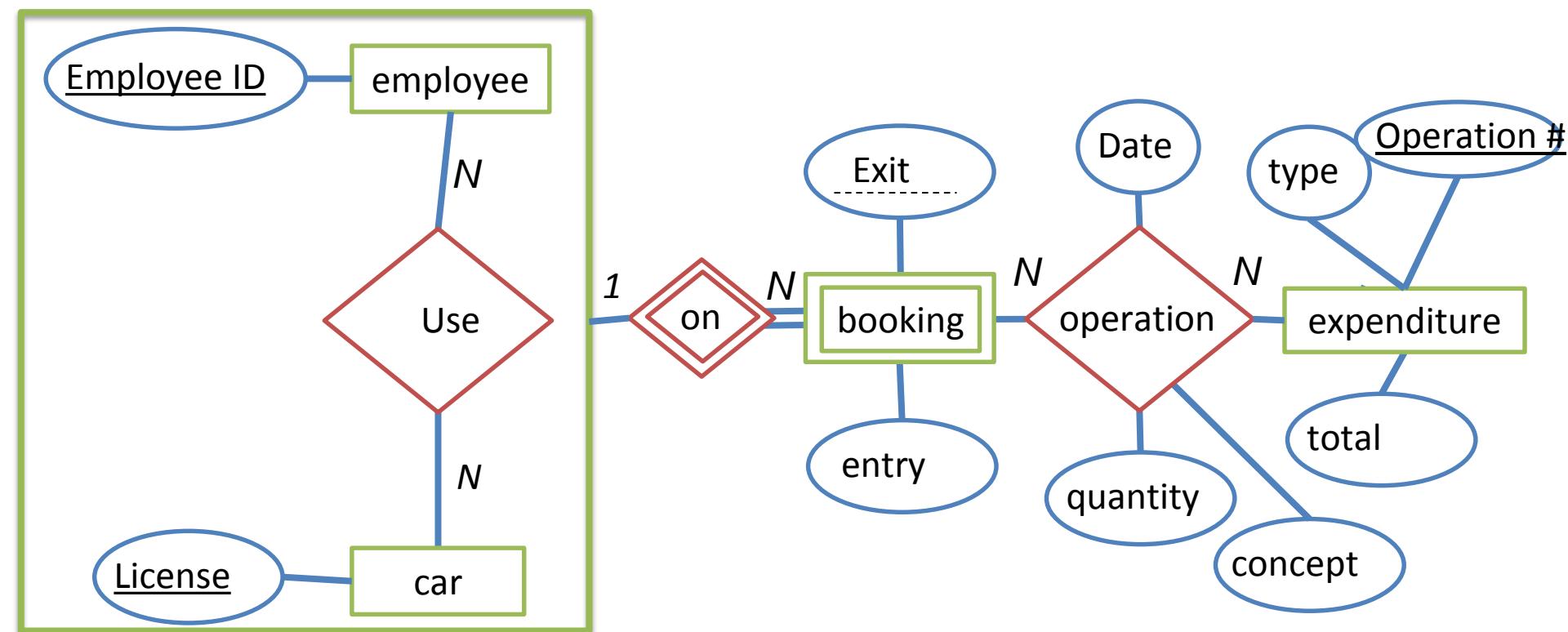
## Example 3: business operations (Solution 1)



Databases

# Aggregation

## Example 3: business operations (Solution 2)



# Block 3

# RELATIONAL DESIGN

Debora Gil, Oriol Ramos, Carles Sanchez

# Contents

1. Introduction
2. ER design to Relational Model

# 1. Introduction

# phases design

Software  
Engineering

BD designer

BD Manager

Database

Requeriments funcionals

Anàlisis Funcional

Especificació de transaccions  
d'alt nivell

Independent del SGBD  
Específic per a cada SGBD

Disseny de programes  
d'aplicació

Implementació de  
transaccions

Programes d'aplicació

Recol·lecció i Anàlisis  
de Requeriments

Requeriments de la base de dades

Disseny Conceptual

Esquema conceptual  
(en un model de dades d'alt nivell)

Disseny lògic  
(Transformació del model de dades)

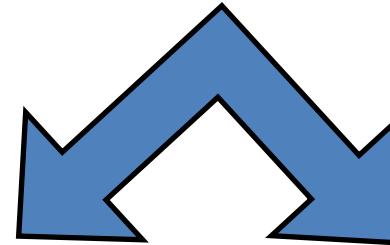
Esquema (conceptual) lògic  
(en el model de dades d'un SGBD)

Disseny Físic

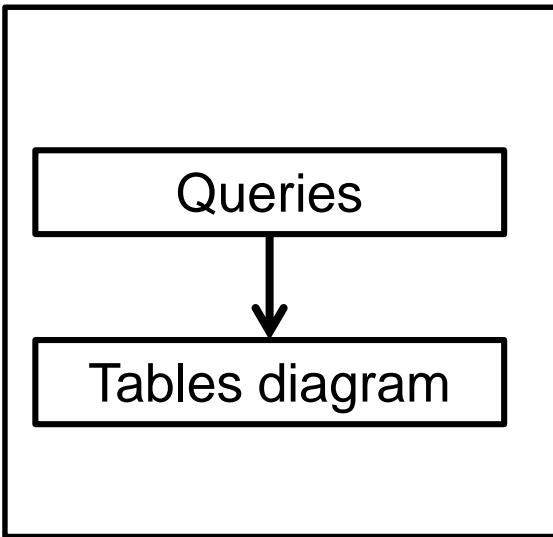
Esquema intern  
(per al mateix SGBD)

# Logical design

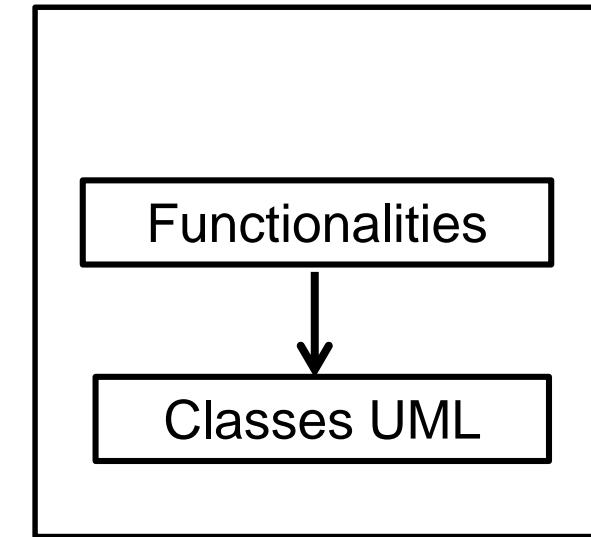
Conceptual Design (ER  
Model)



Databases



Software Engineering



# 1. ER Design Relational Model

1.1 Relational Model

1.2 Tables Diagram from an ER Design

1.3 Examples

# 1.1 Relational Model

# Definition

- Data high level description that guarantees its integrity.
- Is the unique theoretical model implementable with basic rules
- First defined in June 1970 by Edgar Codd, of IBM's San Jose Research Laboratory.
- Has become the predominant type of database, although another models exist.

# Components

**Data structure.** Description by Relations / Tables.

**Integrity rules.** Rules to ensure the queries consistency

Entity (Table, PK)

Referential (relations between tables, FK)

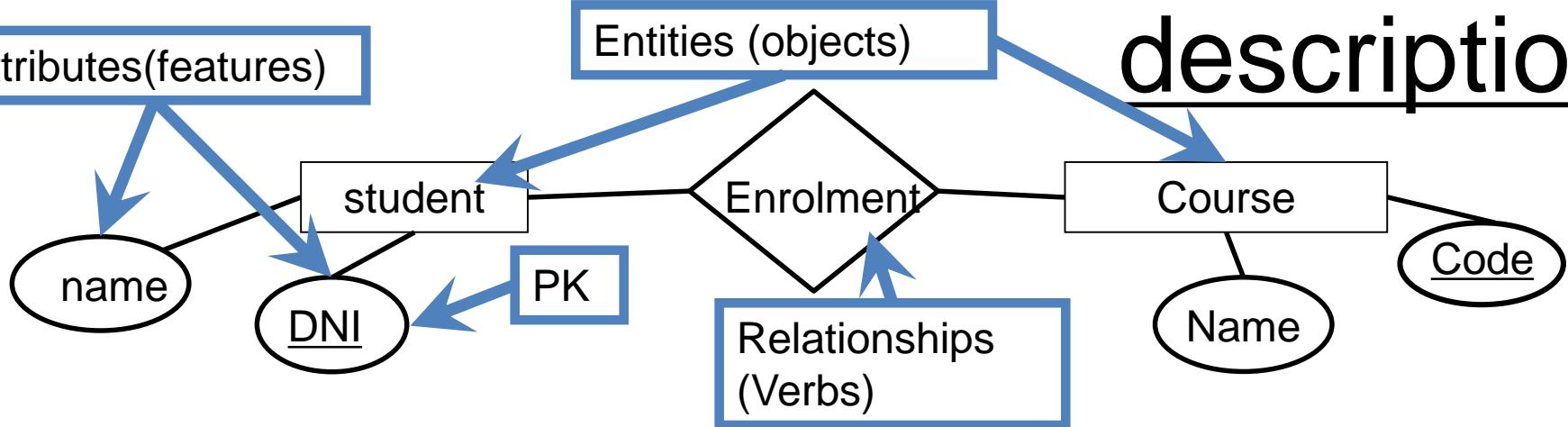
---

**Operators** to access data by content:      Relational algebra

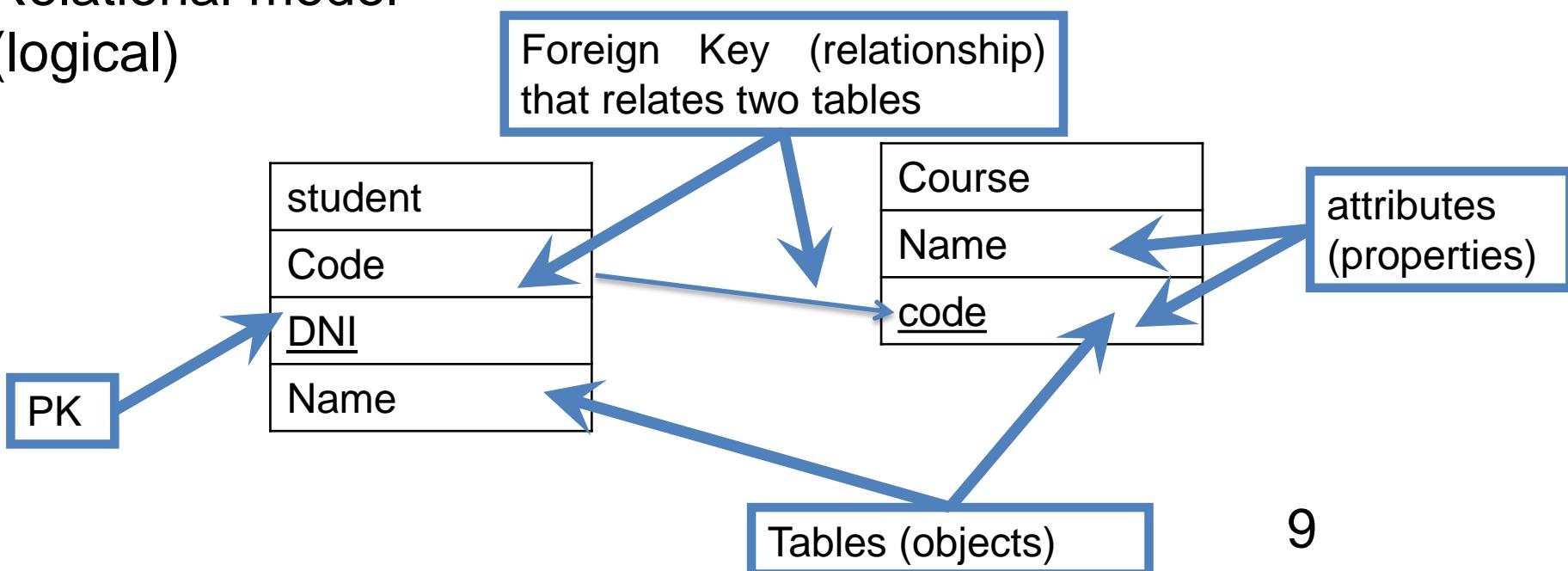
Relational calculus

## ER Diagram (abstract)

# Data description



## Relational model (logical)

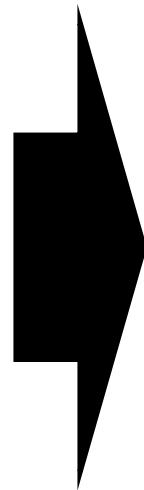


# Relational components

Objects

Features

Related information



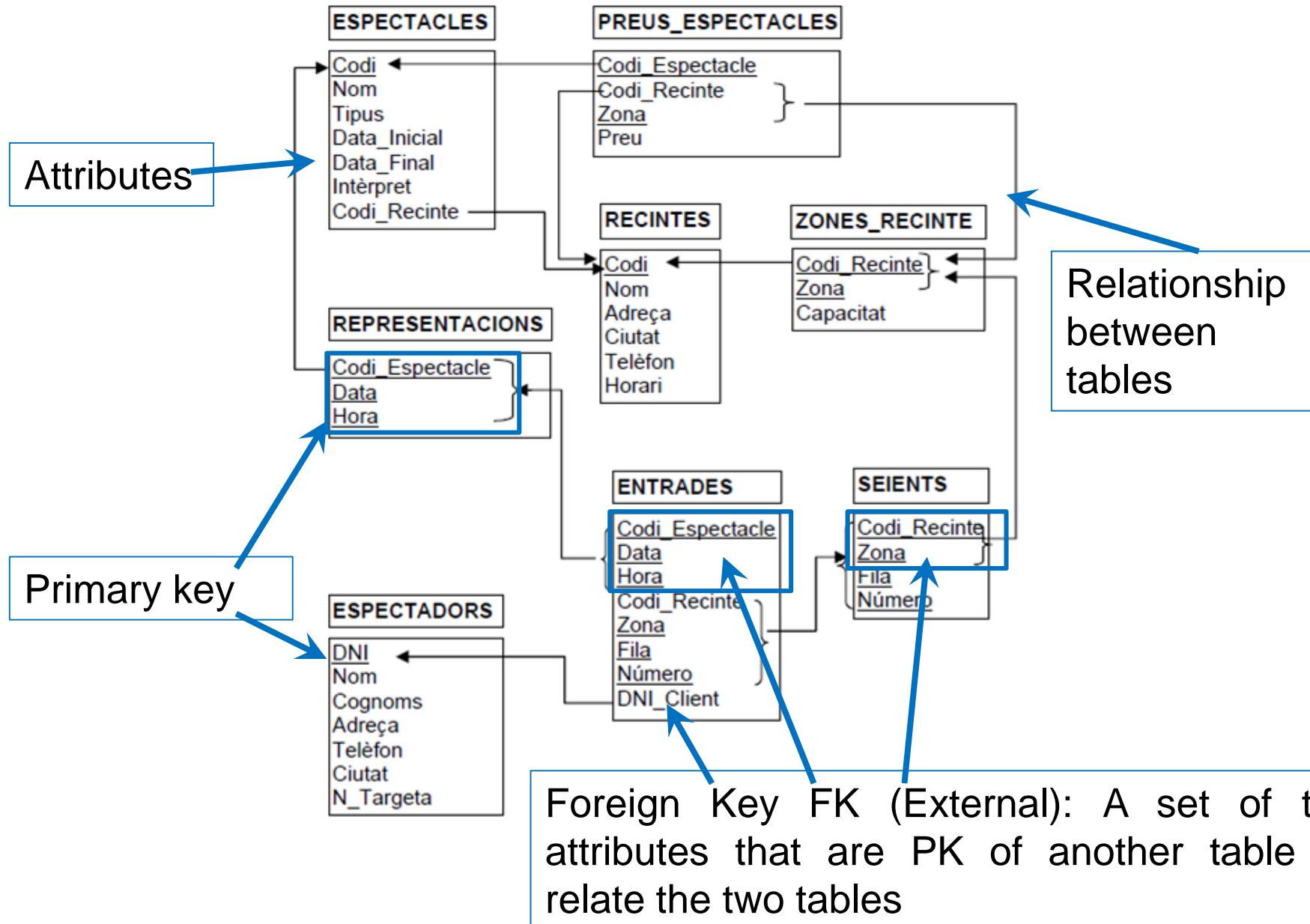
Relation (Table)

Domain (allowed values of  
attributes)

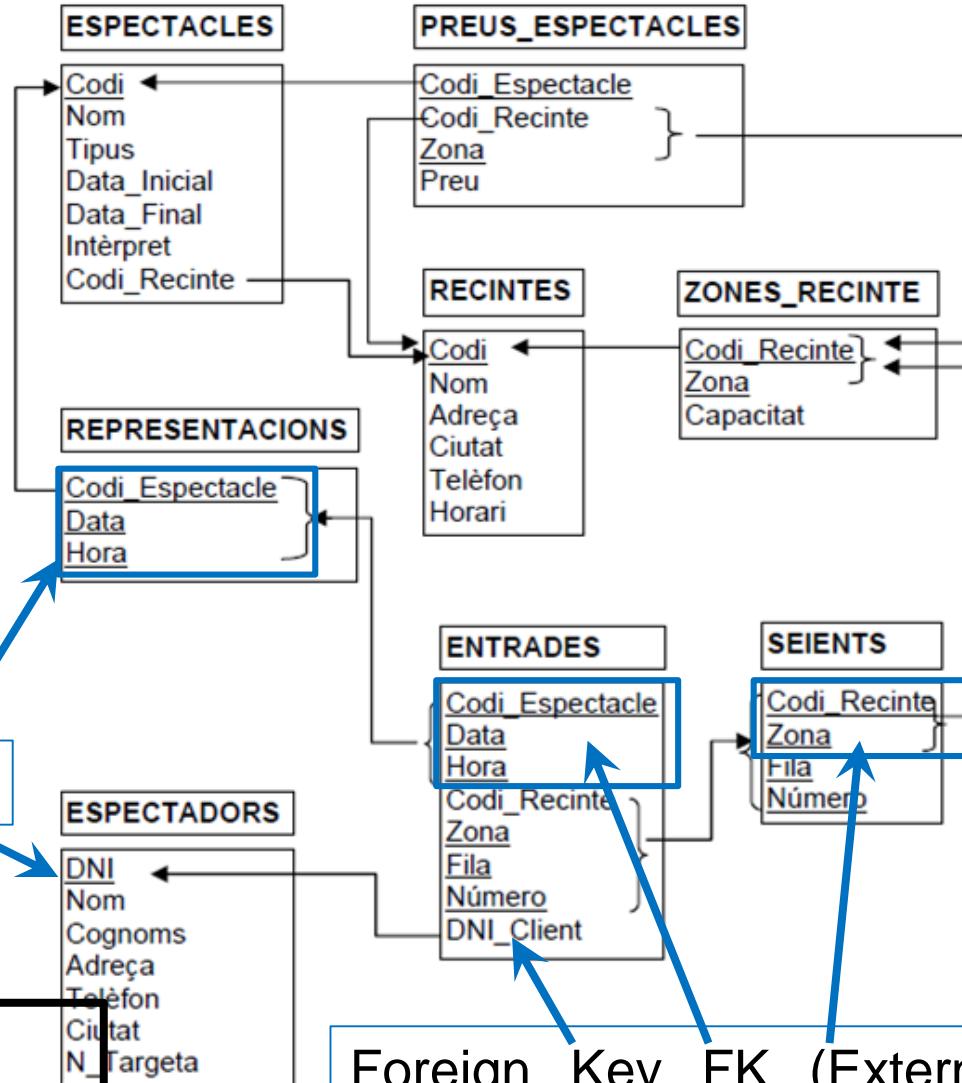
Relational database (set of  
referenced tables)

# Database

Collection of related tables  
(referenced, linked) together



# Database

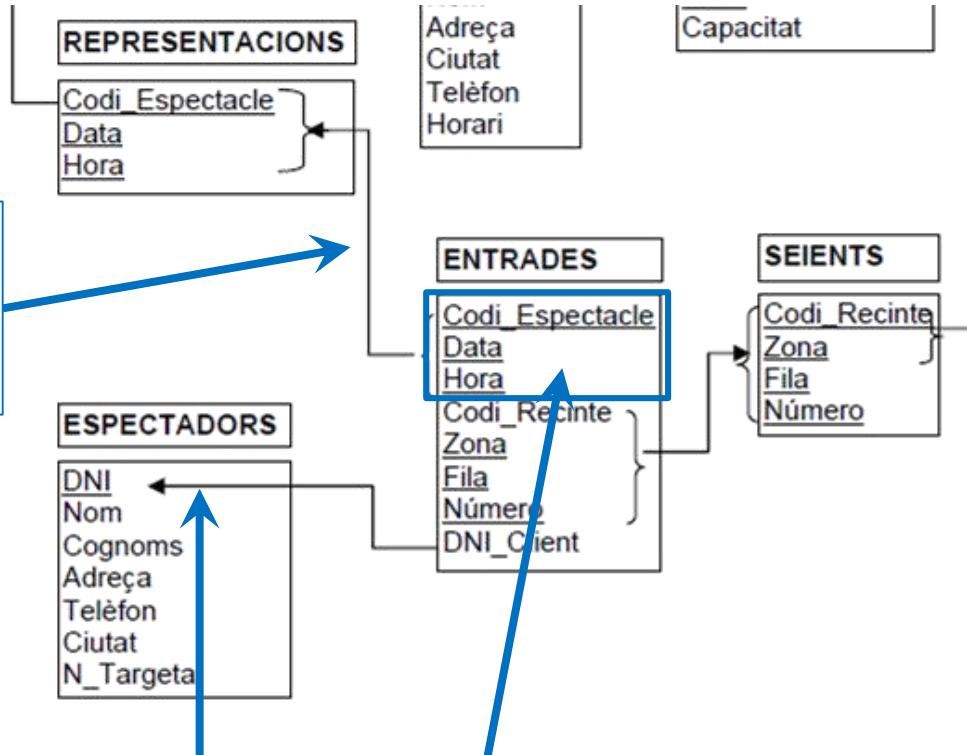


Primary key

They are  
essential for  
integrity

Foreign Key FK (External): A set of table  
attributes that are PK of another table and  
relate the two tables

# Database Integrity



Relationship  
between tables (FK  
referencing to a PK)

FK contains all the PK attributes of the PK referred (pointed to)  
The arrow must always point to PK to ensure propagation  
changes

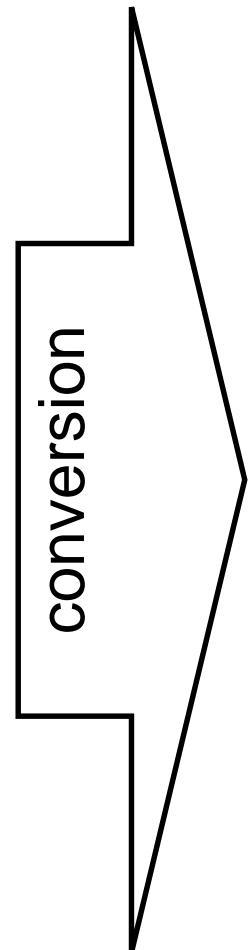
# 1.2 Tables diagram from an ER Design

# Structures

## ER design

- Entities      {
  - Strong
  - weak
- Relations      {
  - n-n
  - 1-1
- Attributes      {
  - Simple
  - Multivalued
  - PK
  - Composite
- Aggregations
- Specializations

## Tables diagram



- Table (header with the attributes, PK)
- Foreign Key FK (references)
- Attributes (Simple)

# Conversions

- Entities → Table

Strong: Entity PK      Weak: Weak entity PK + FK with value strong PK

- Relationships

- **Binary 1-n:** FK on n-side entity with PK value on 1-side entity  
Ex: Table A - PK A                  Table B - PK B + FKA
- **Binary 1-1:** FK on 1-side entity with PK value on 1-side entity
  - + FK uniqueness constraint

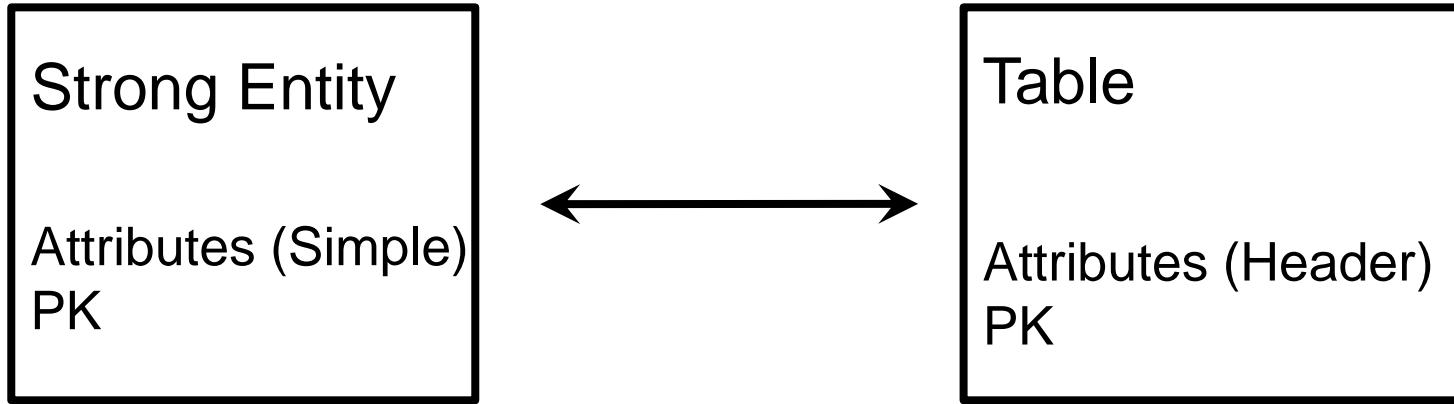
Ex: Table A - PK A + FK B    Table B - PK B + FKA
- **Binary n-n:** Table with PK union participating entities PK's +FK for each participating entity  
Ex: Table A - PK A    Table B - PK B    Table C – PK( FK A + FK B)
- **Ternari:** Table with PK union n-side participating entities PK's +FK for each participating entity

- Attributes

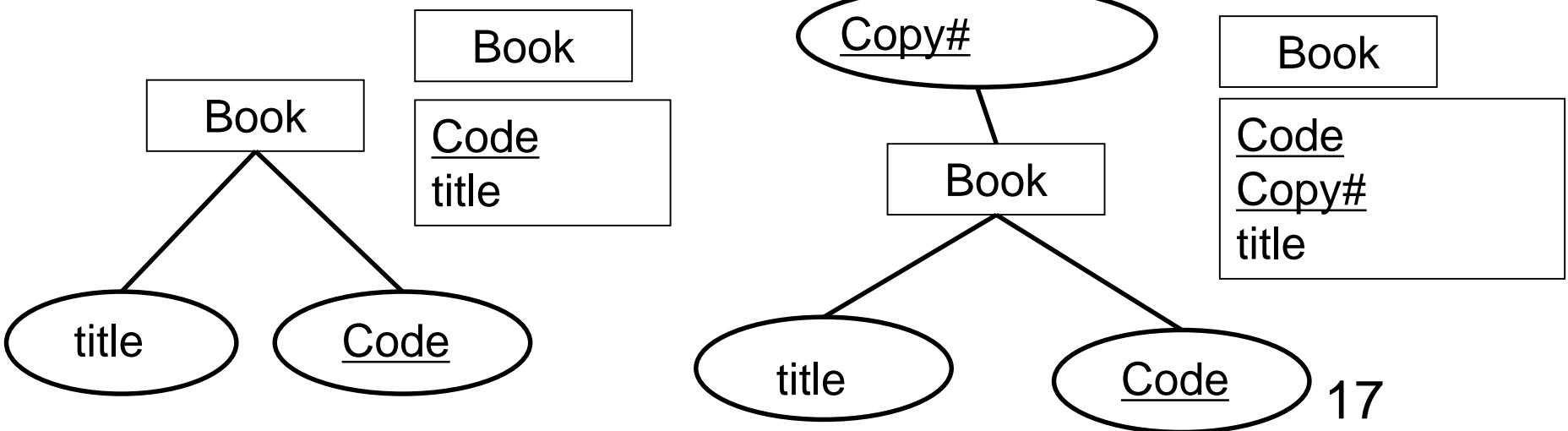
Simple, composite → attributes

Multivalued: Table with attribute PK and entity PK + FK with value entity PK

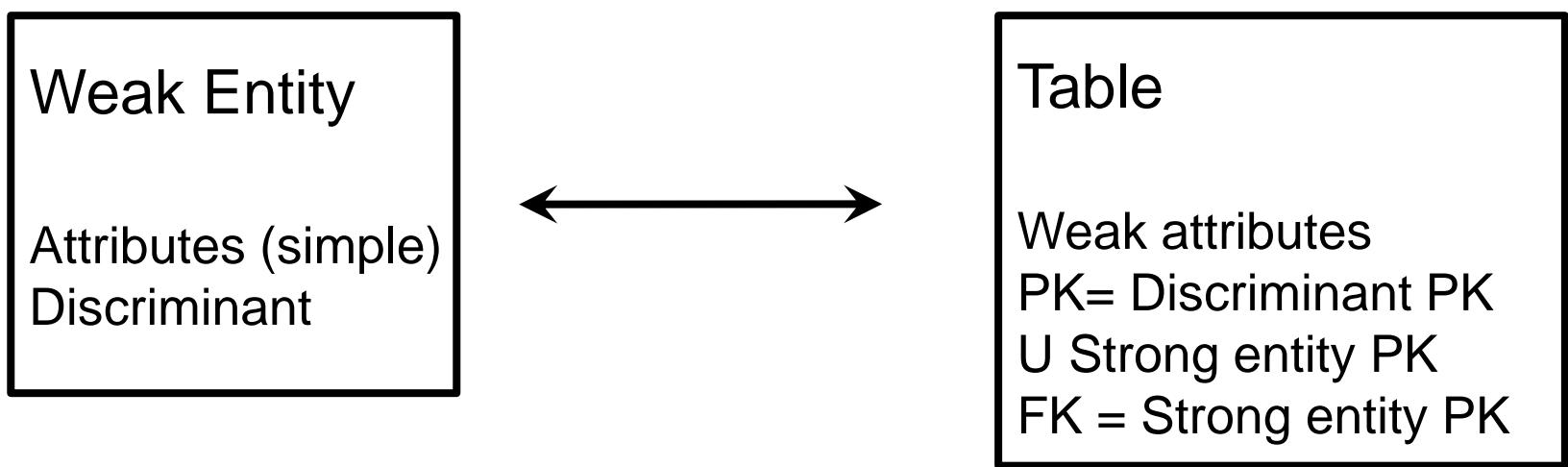
# Strong Entities



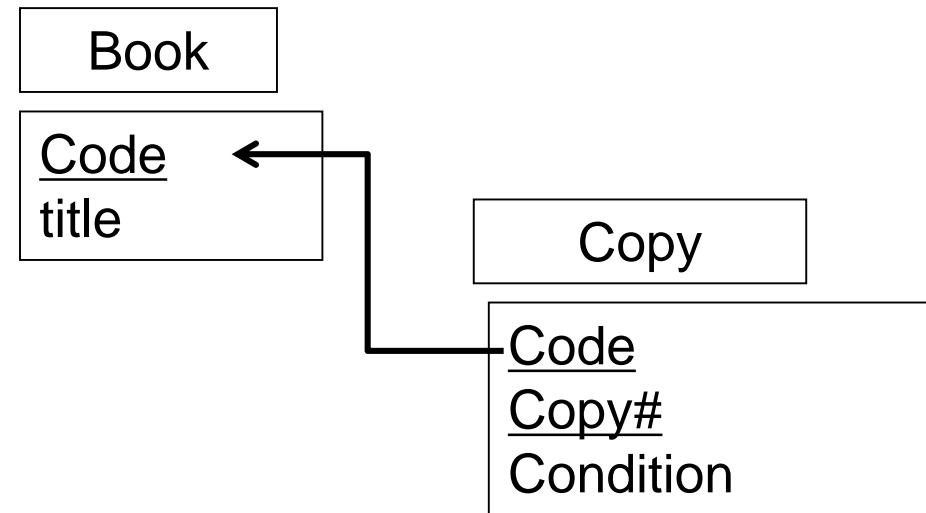
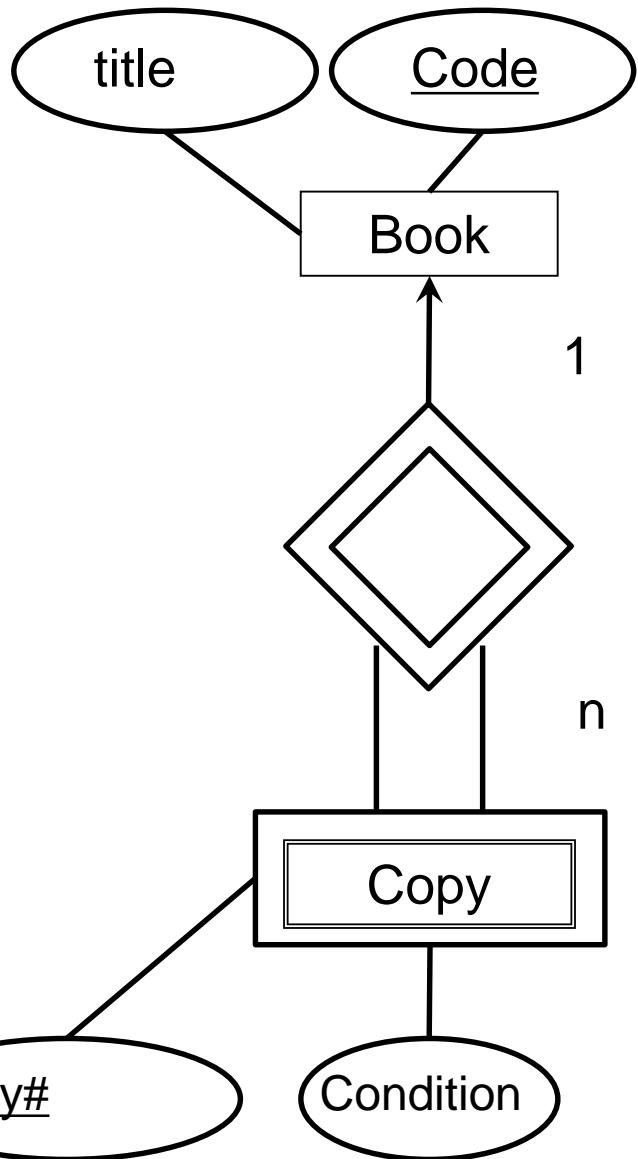
Example:



# Weak Entities



# Example



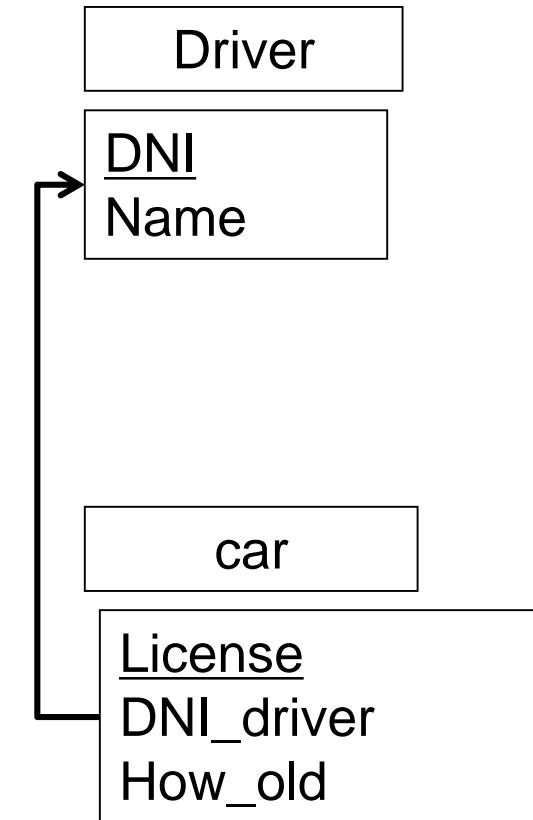
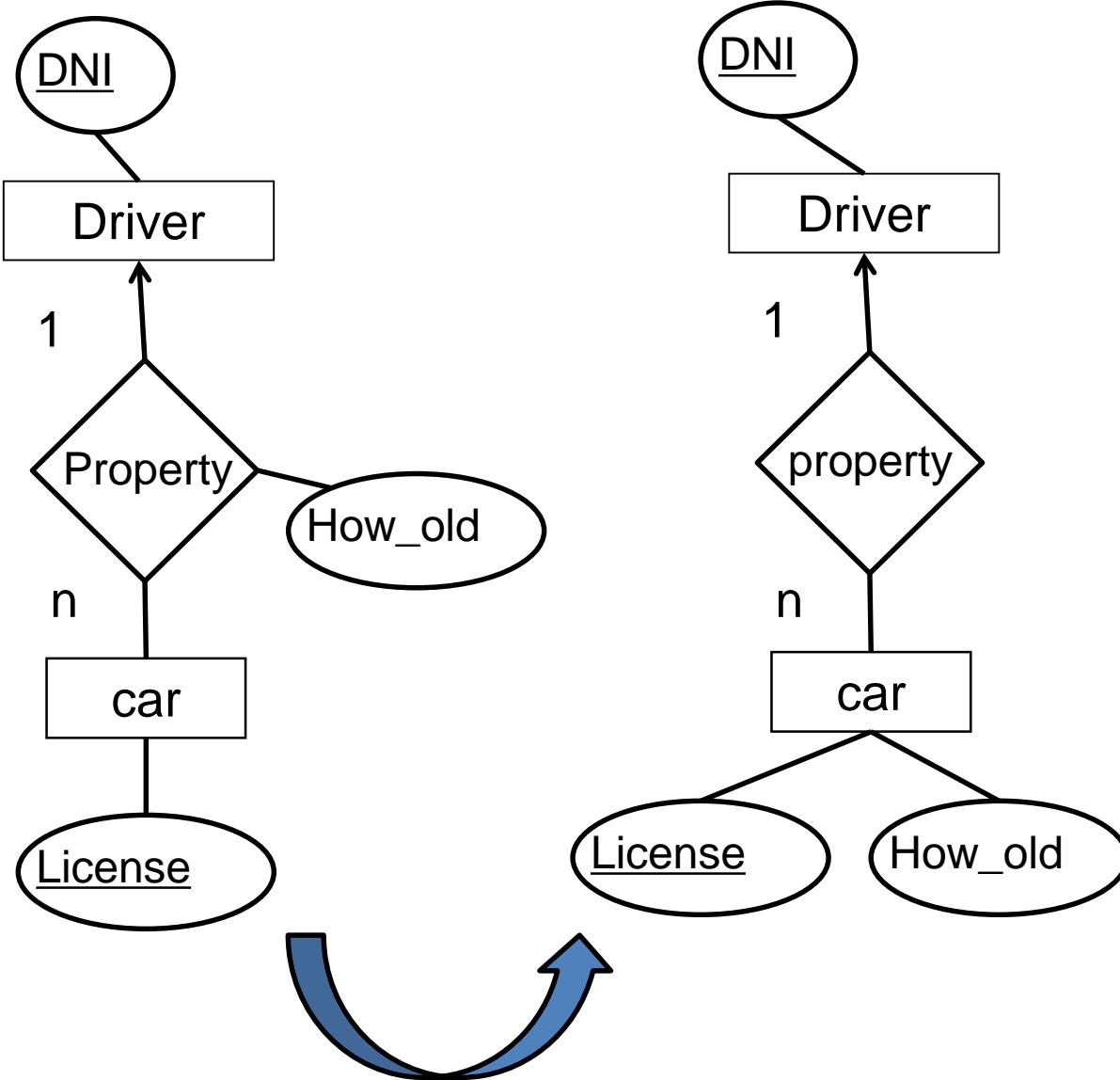
# Relationships

They become a table or a relationship between interrelated entities depending on the cardinality:

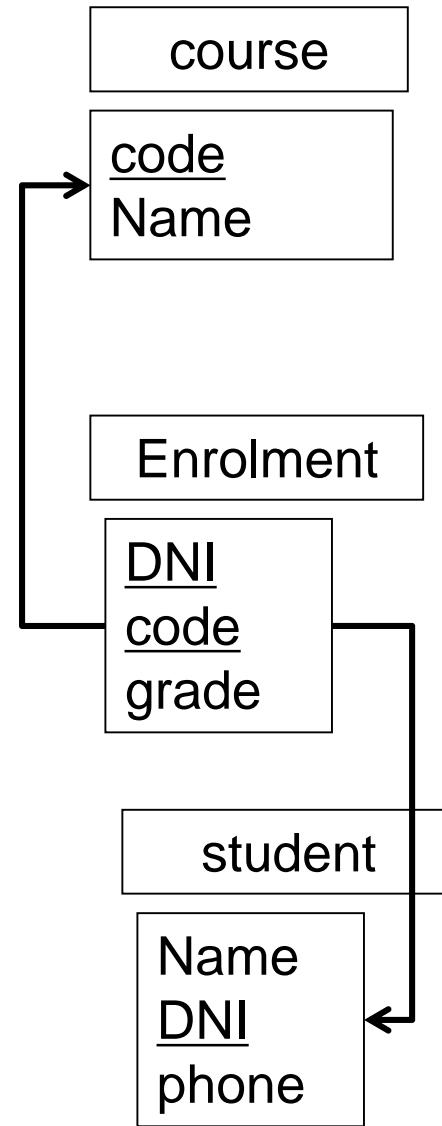
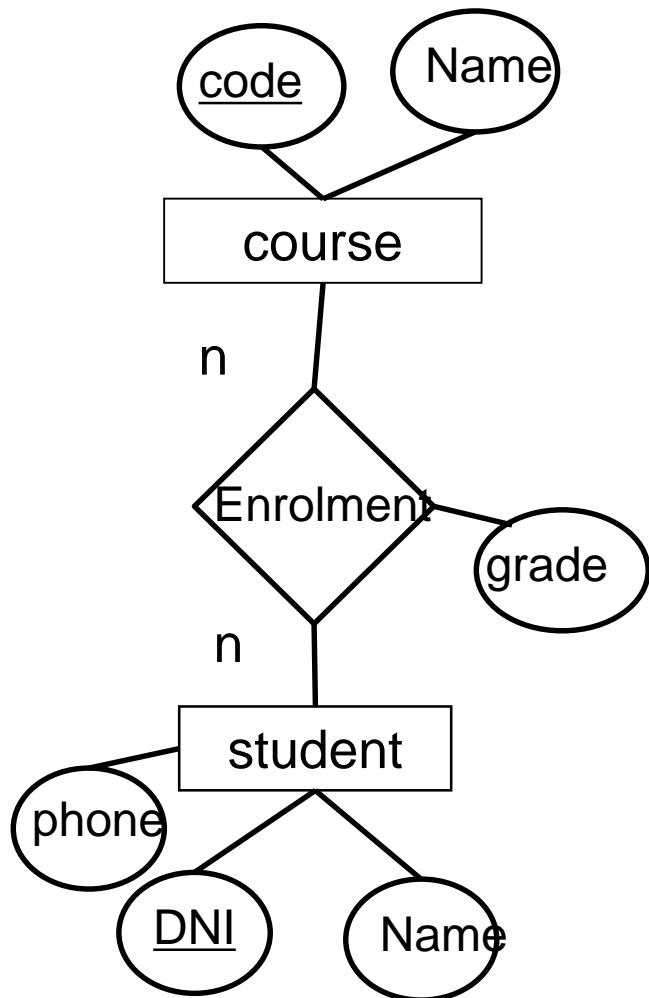
Cardinality 1-n: Attributes are passing to n-side entity and the interrelationship is converted to relationship between tables (FK to the n-side entity)

Cardinality n-n: Relationship is converted to a new table with FK linking two related tables. The new table PK is the unión of the related tables PK

# Example. cardinality 1-n



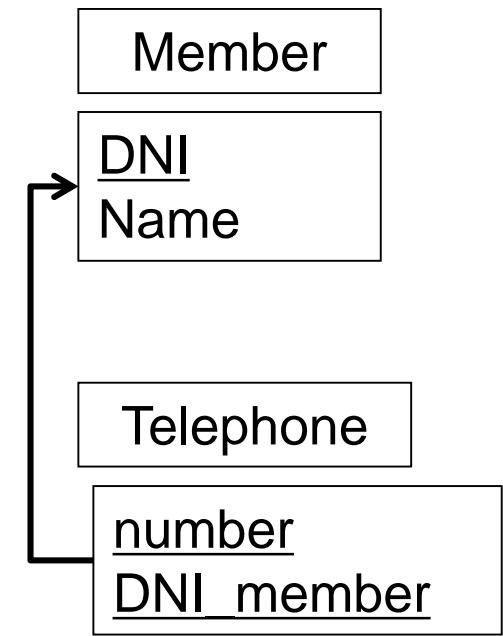
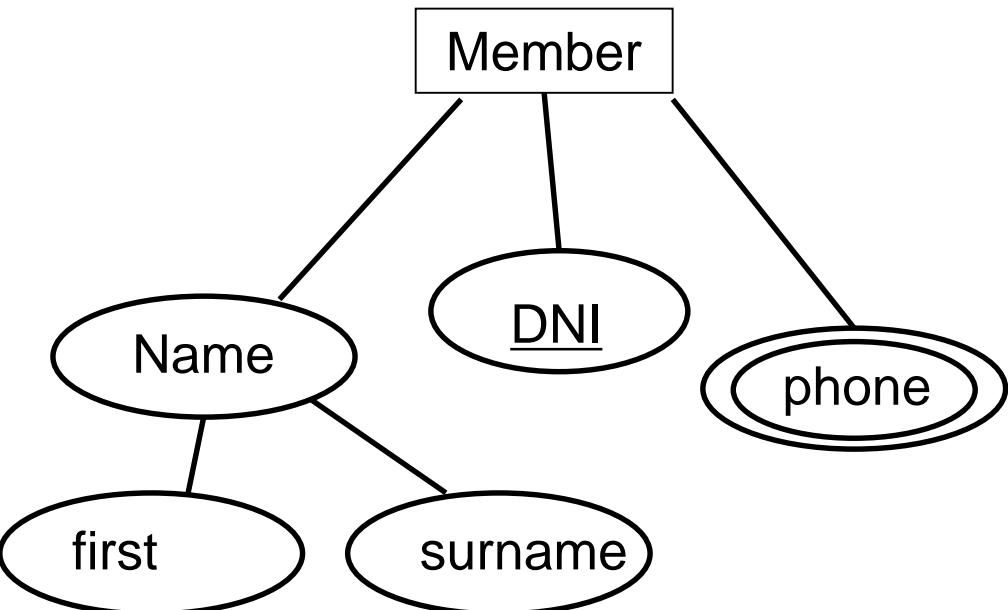
# Example. cardinality n-n



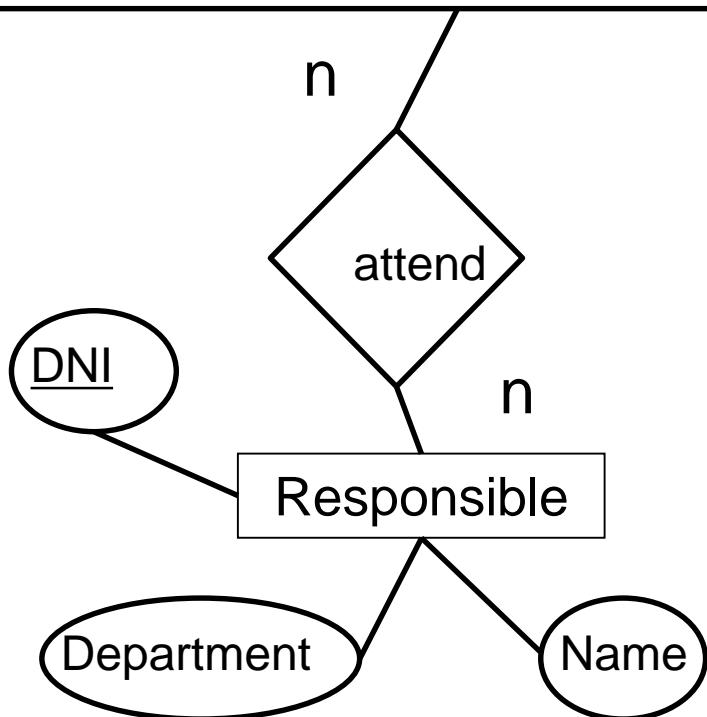
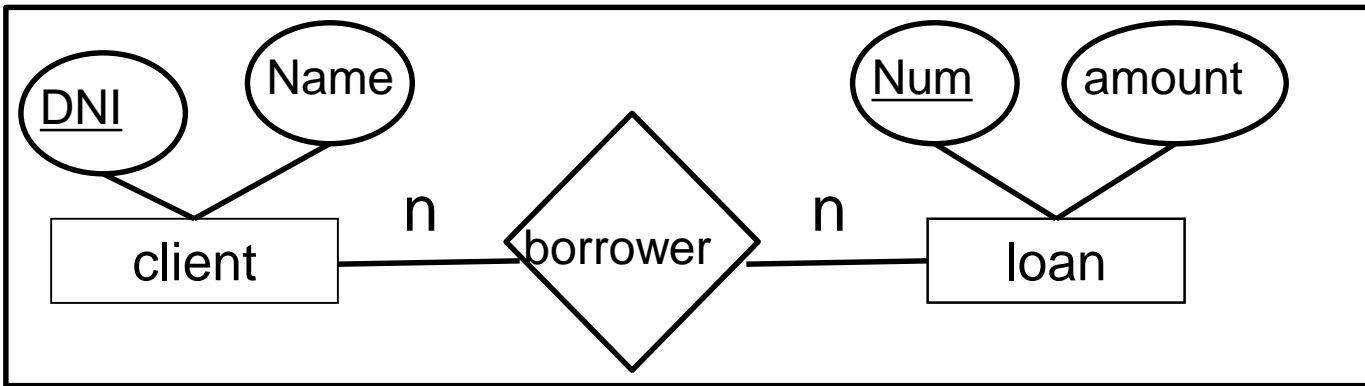
# Attributes

Simple and composite attributes have direct translation into simple attributes with a certain domain.

Multivalued generate a table with FK:

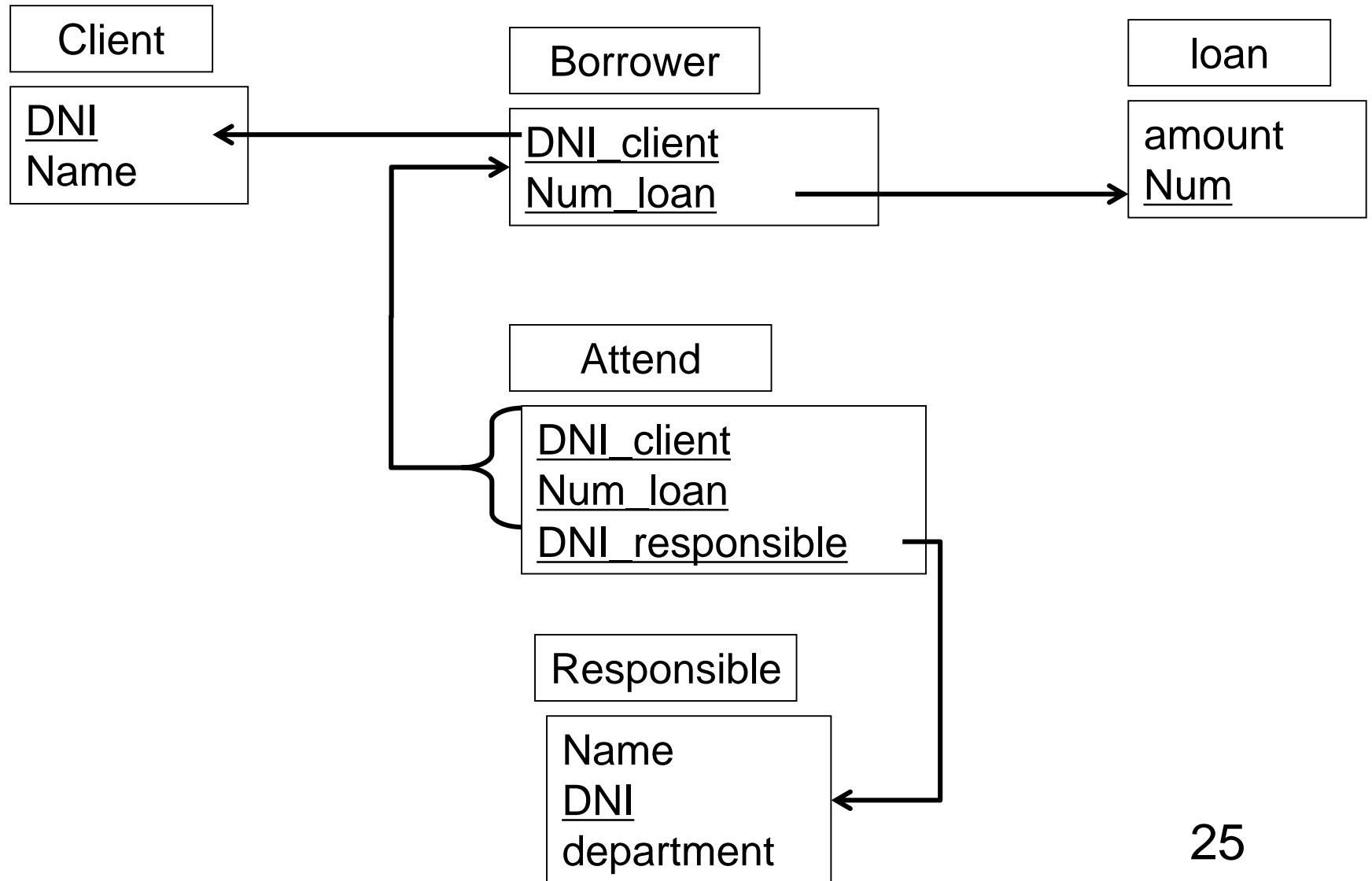


# Aggregations

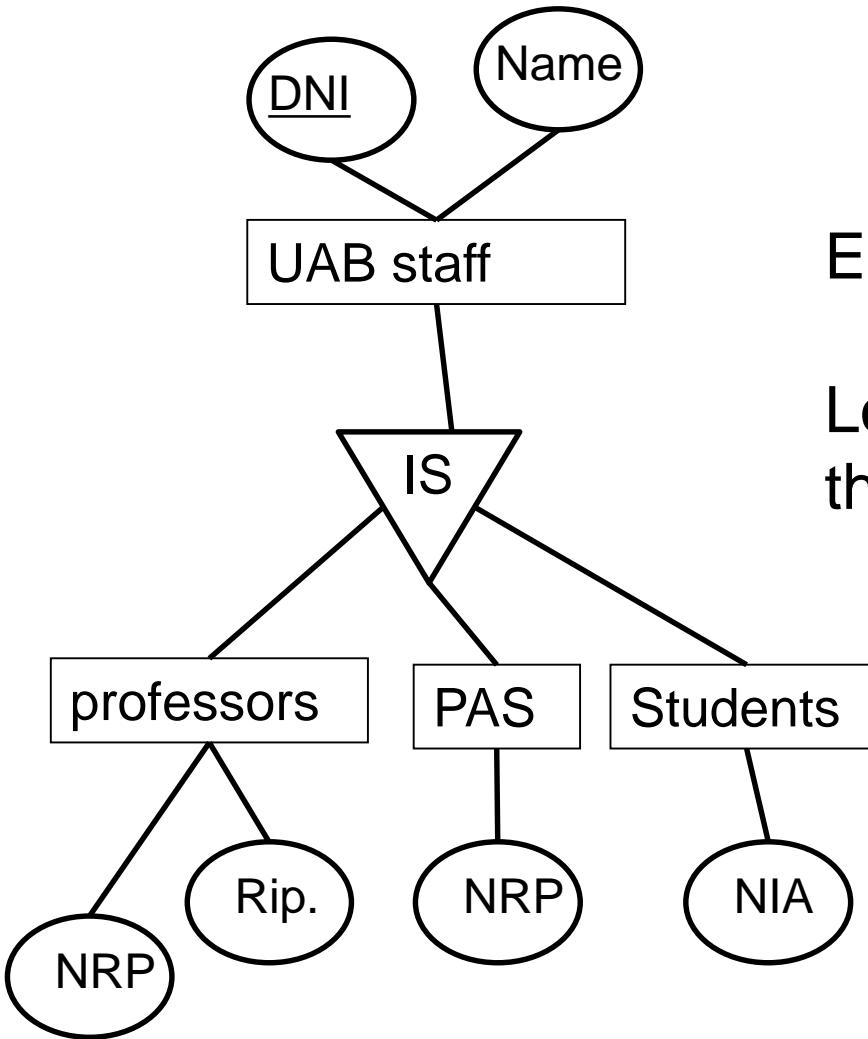


Each n-n relationship becomes a table with FK establishing the corresponding links

# Example



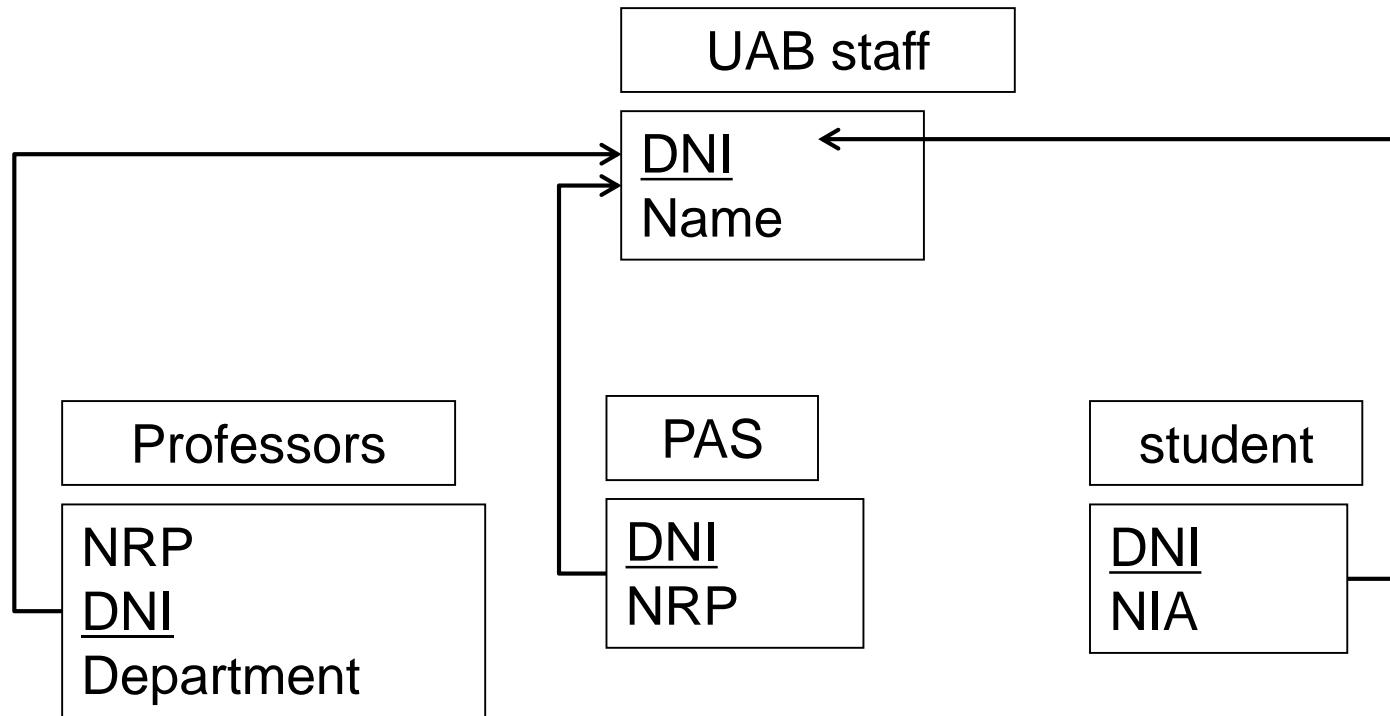
# Specializations



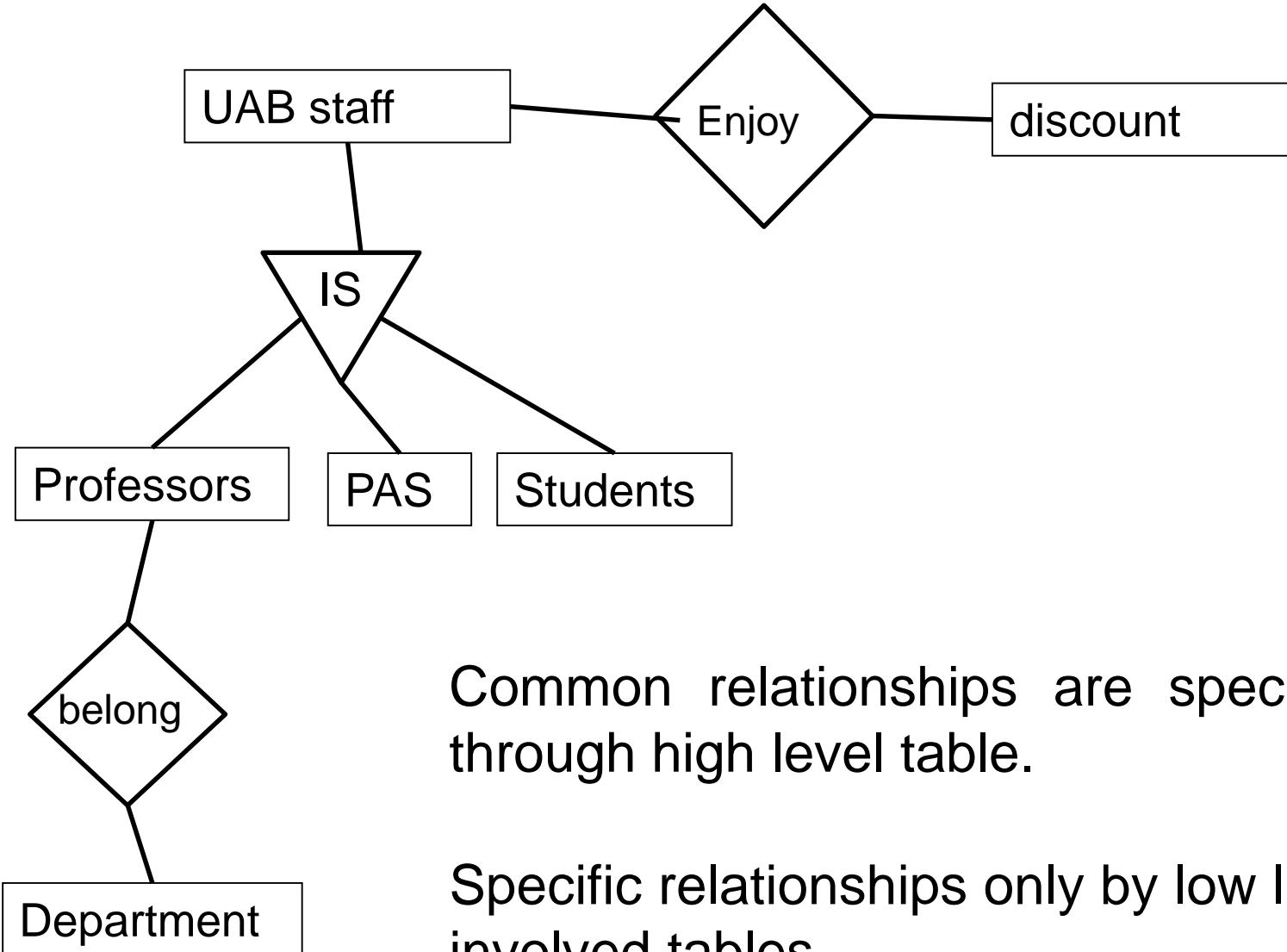
Each entity becomes a table.

Low level entities have FK linking them to high level entity

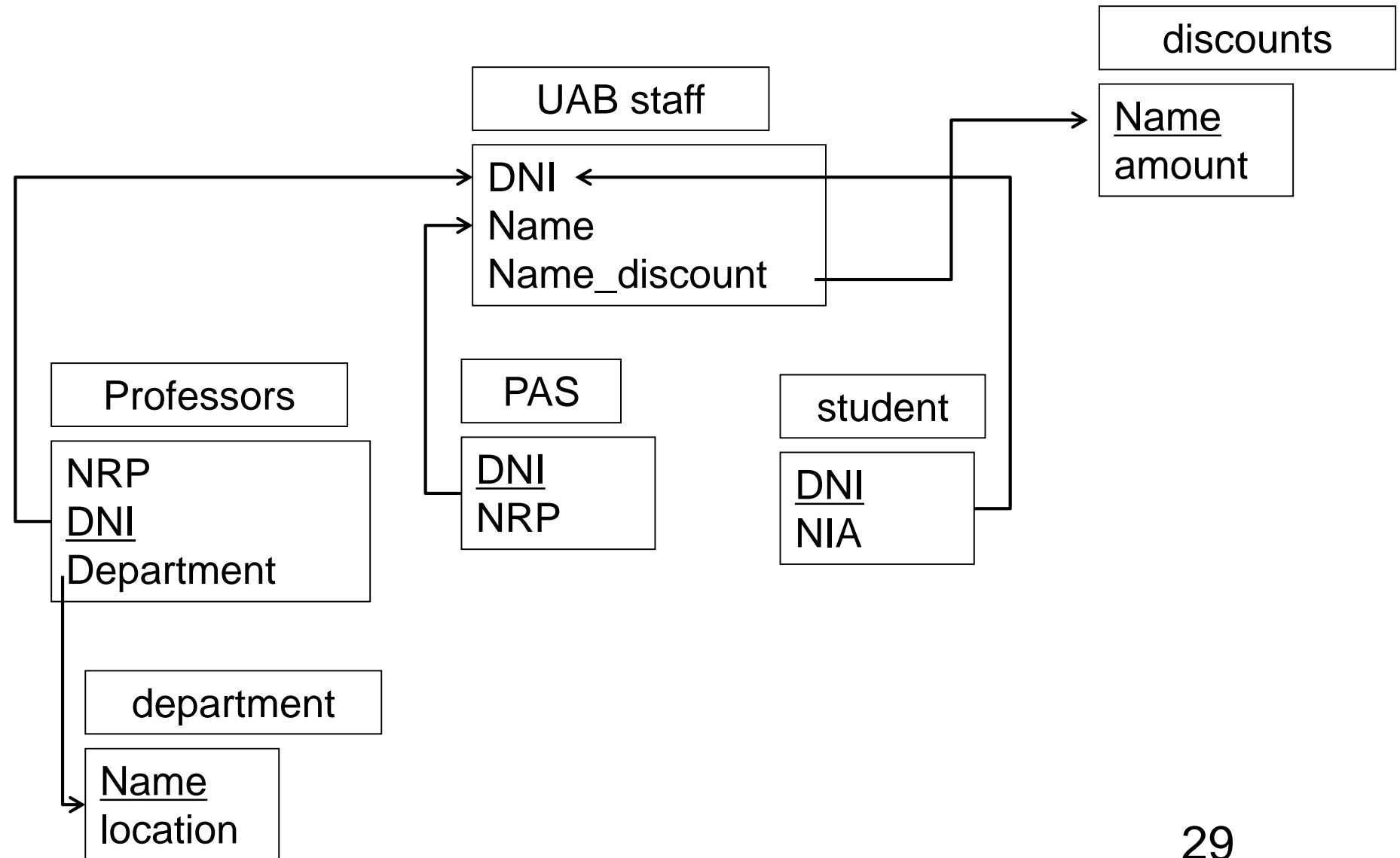
# Example



# Specializations

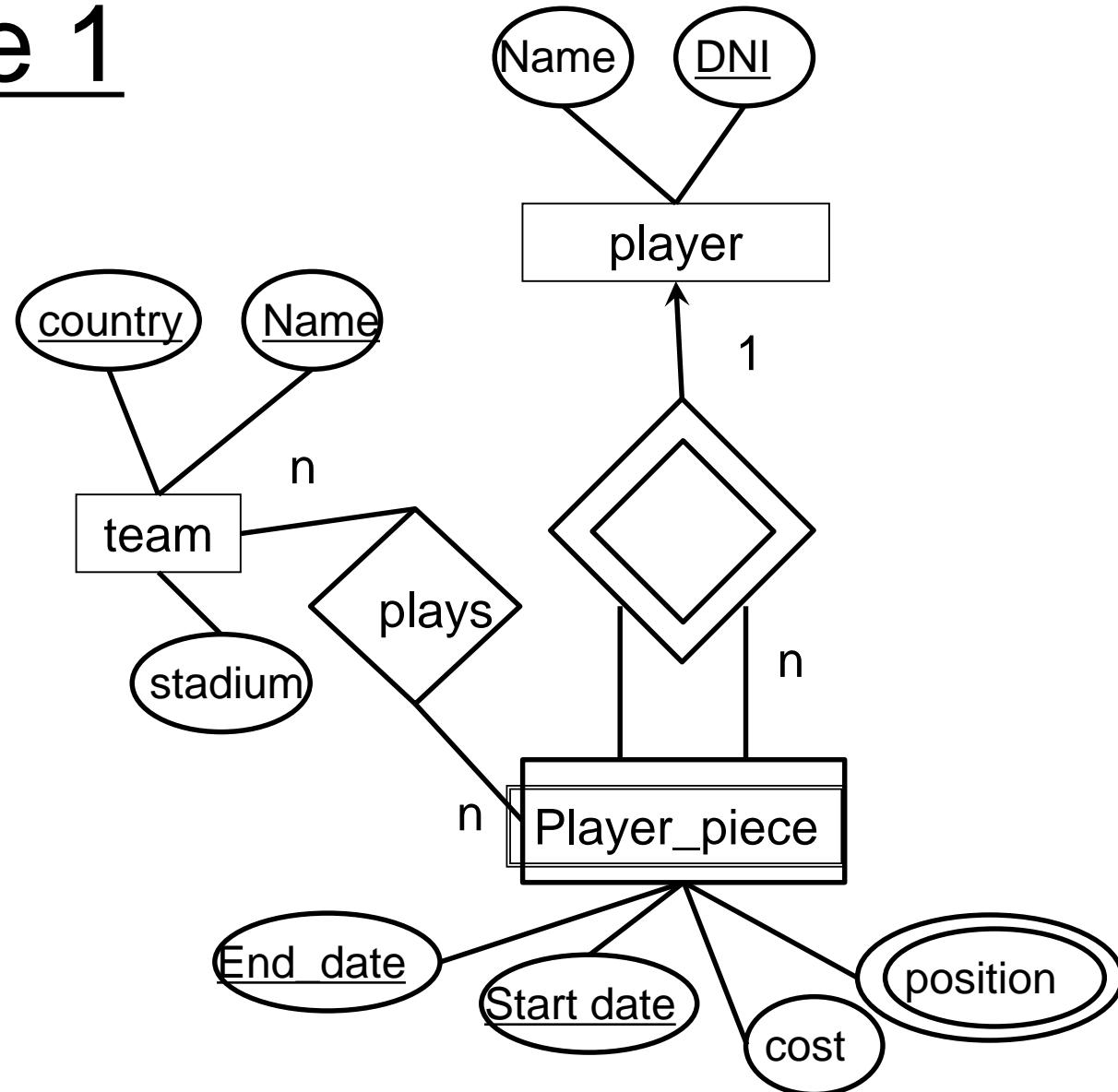


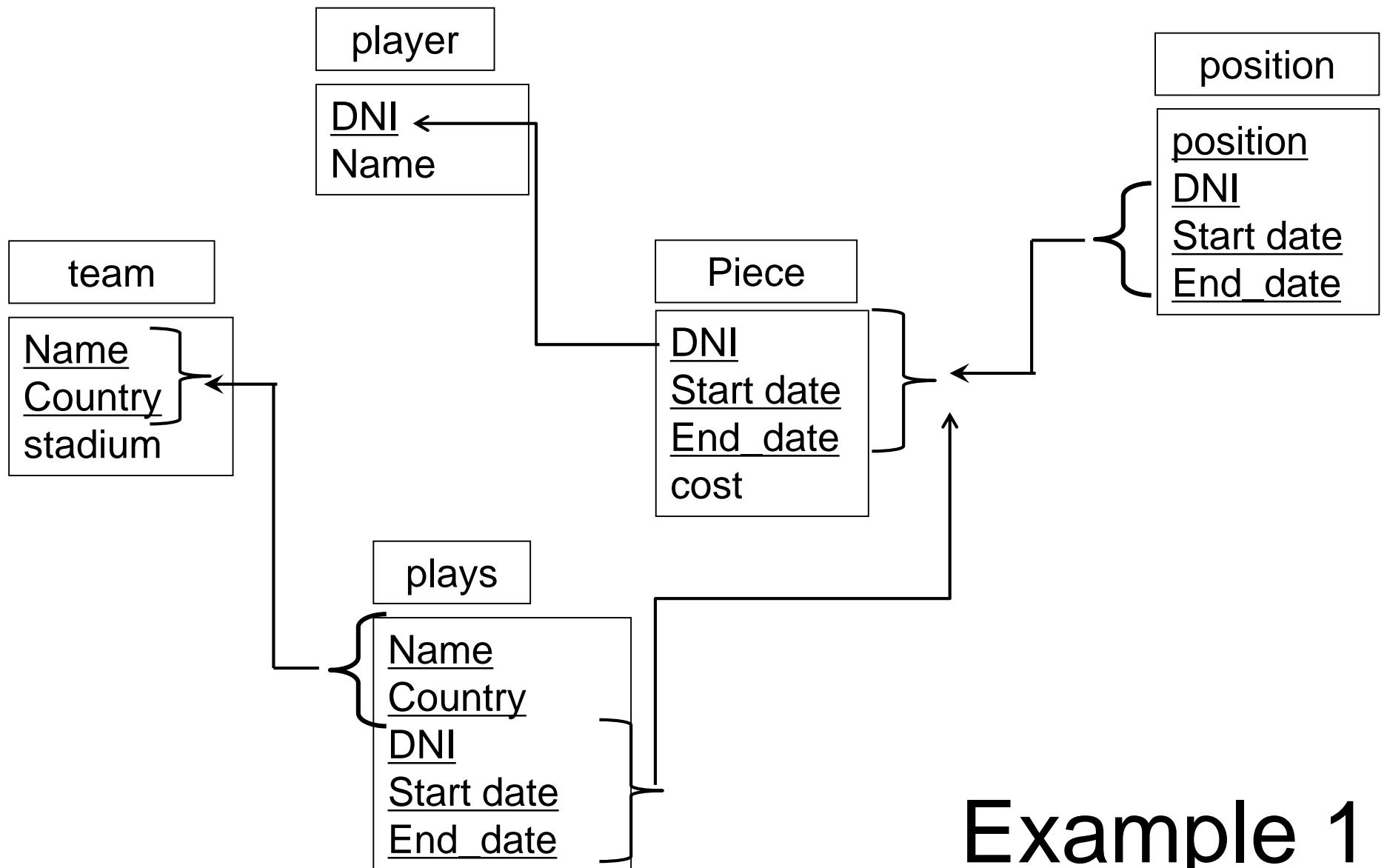
# Example



## 1.3 Examples

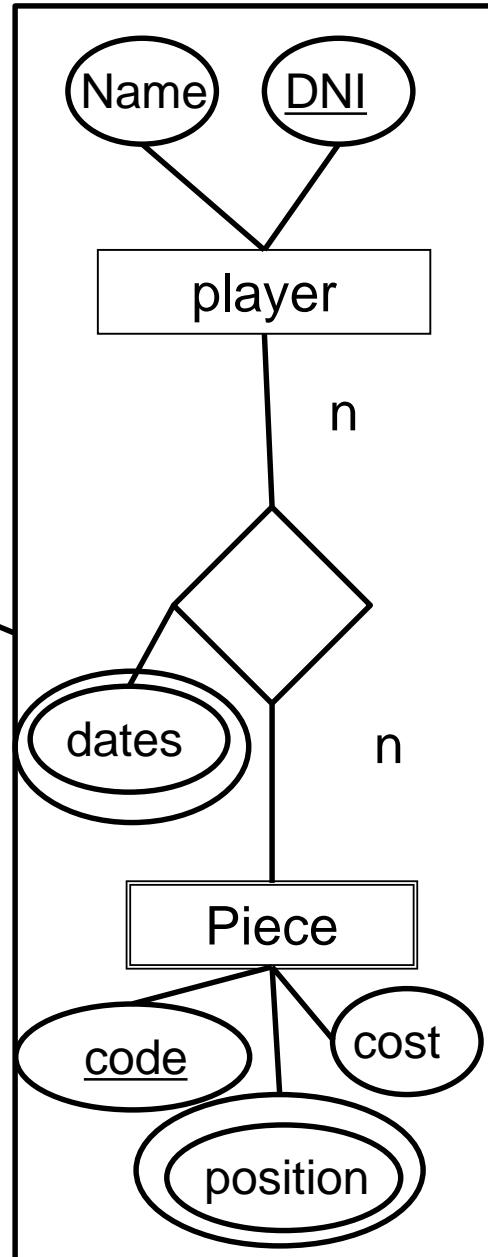
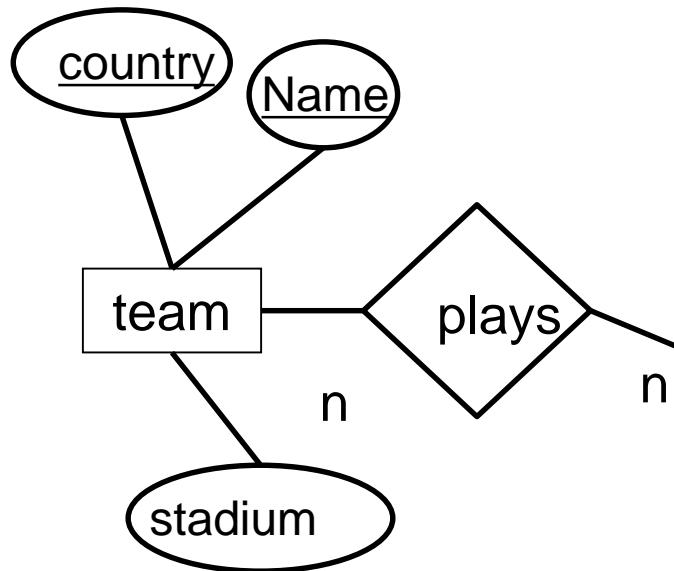
# Example 1

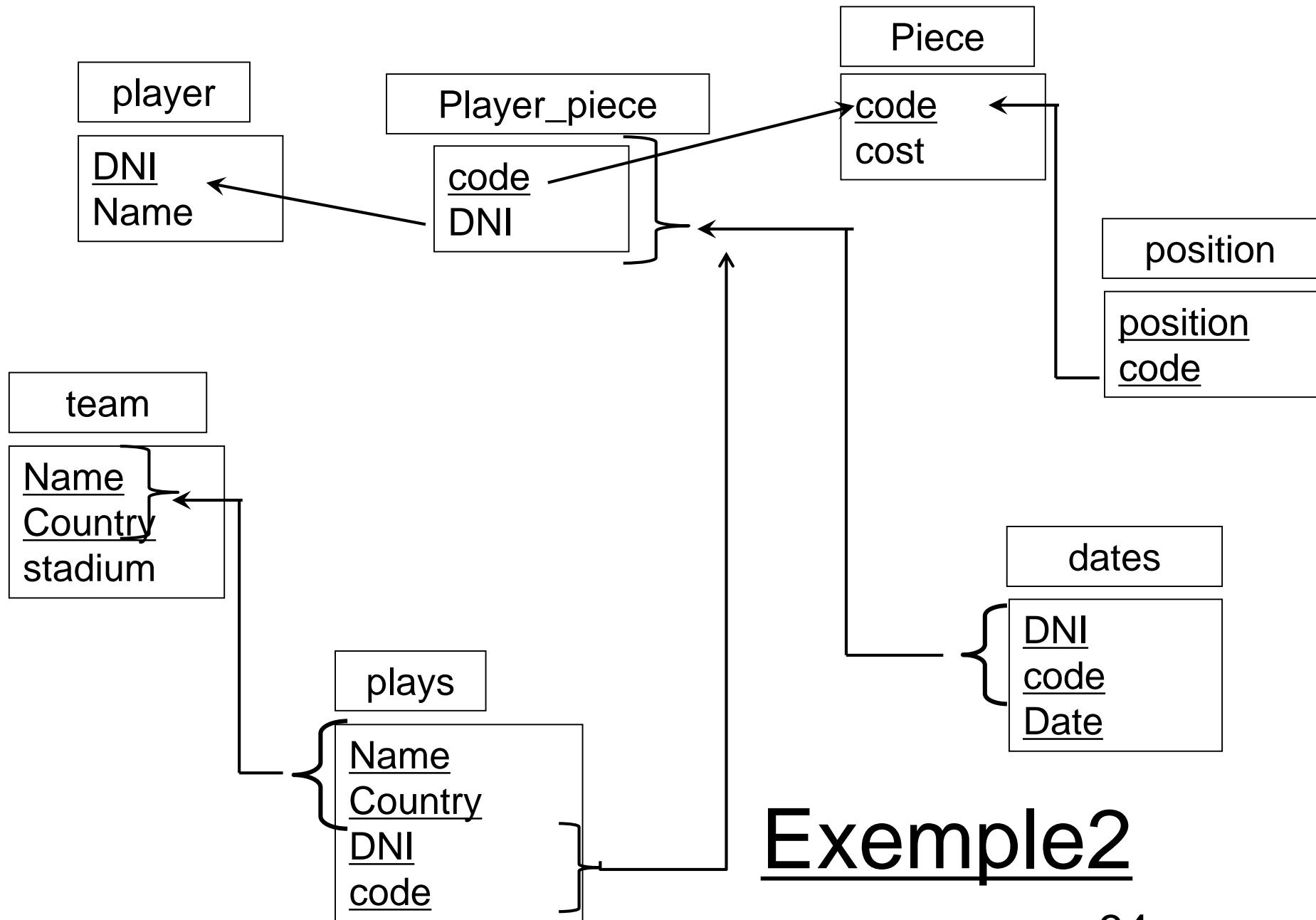




# Example 1

# Example 2





## Exemple2

# Block 3

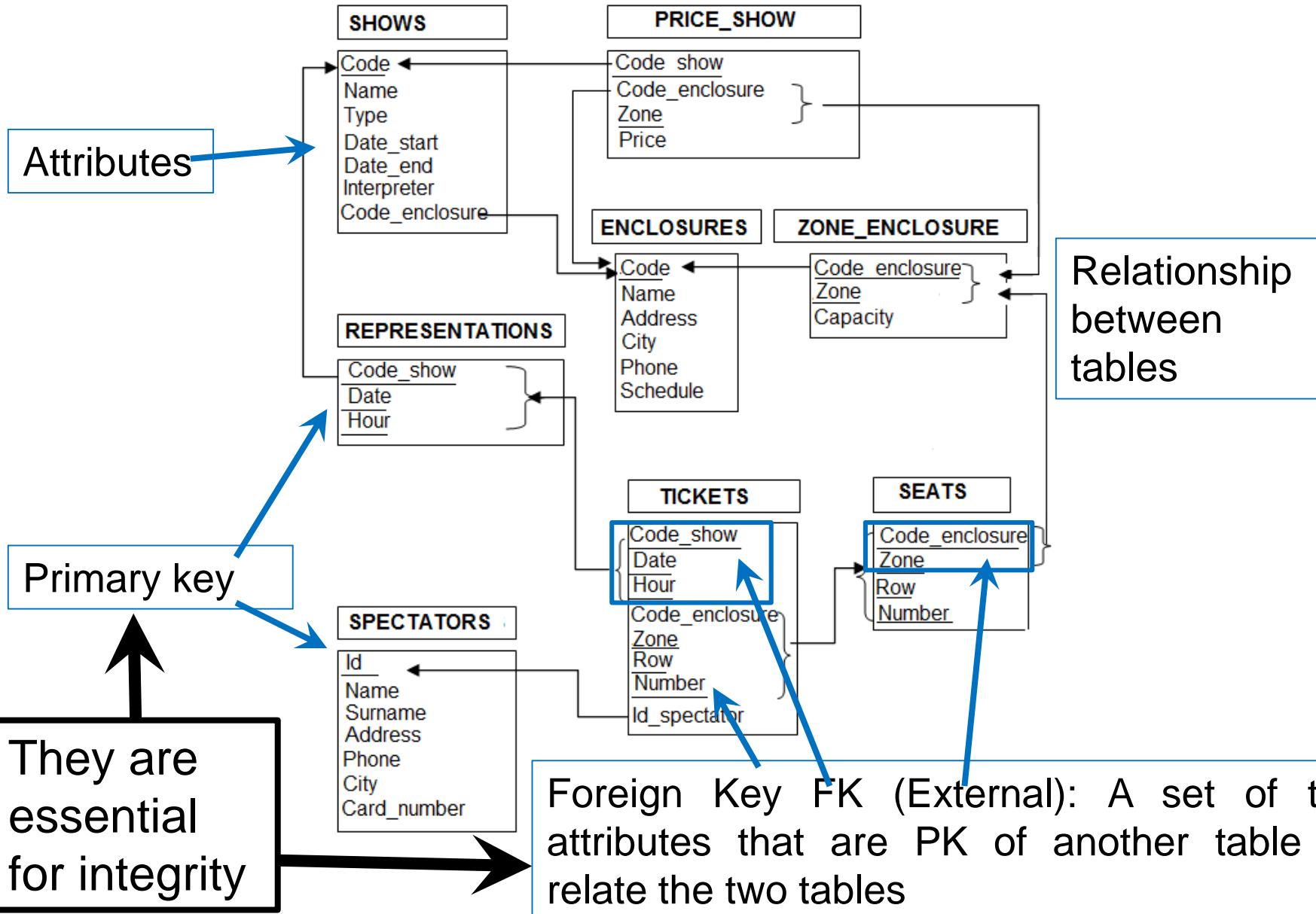
# DATA INTEGRITY

Debora Gil, Oriol Ramos, Carles Sanchez

# 1 Relational Database

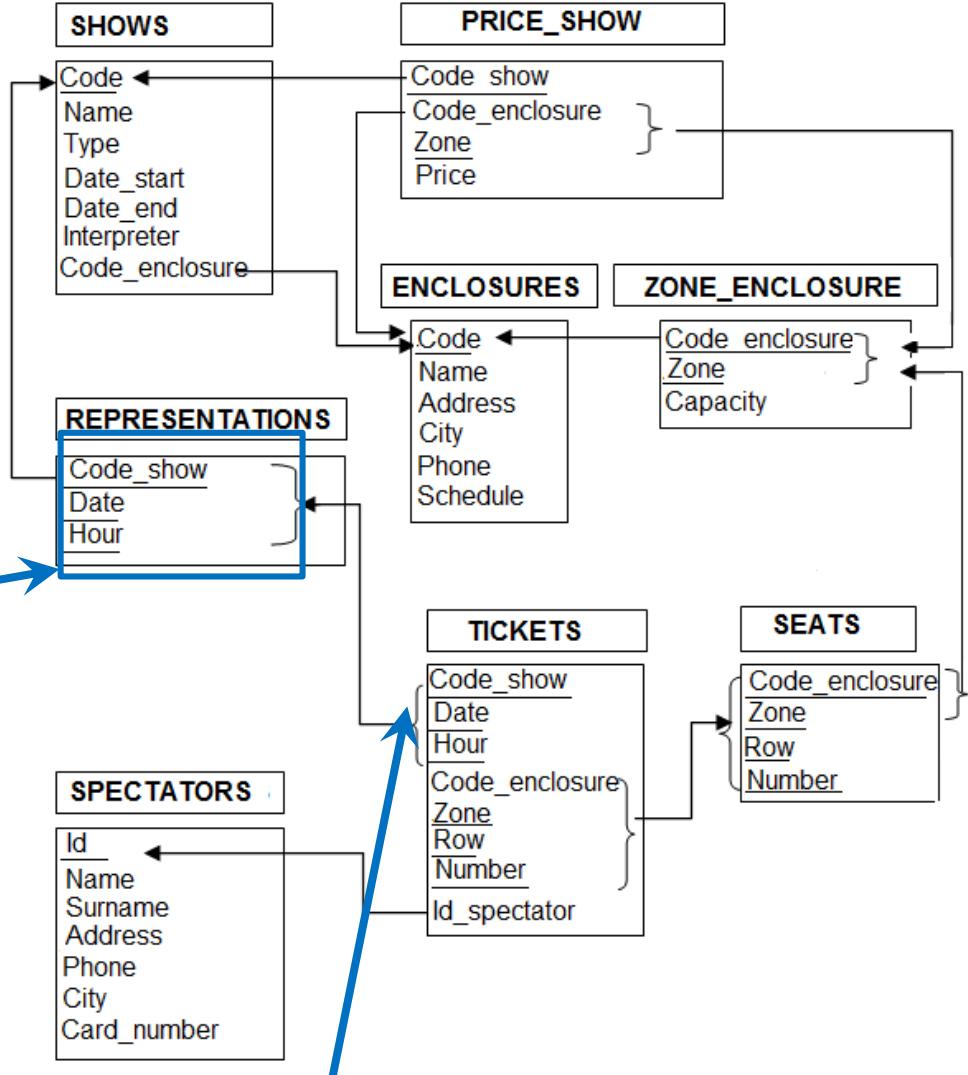
# Database

Collection of related tables  
(referenced, linked)



# Database Integrity

Relationship  
between tables (FK  
referencing to a PK)



FK contains all the PK attributes of the table referred  
(pointed to)

The arrow must always point to PK to ensure propagation of  
changes

## 2. Integrity Rules

2.1 Basic Concepts

2.2 Domains

2.3 Relationships

2.4 Referential

## 2.1 Basic Concepts

# Definition

Rules (constraints) used to ensure validity and accuracy of a data set.

Most are incorporated into the DBMS

Two Levels:

- Specifics
- Generals

# Specific Rules

- Specific to the type of data that each DB contains.
- They are defined through specific domains.

Examples:

- Weight is positive
- Boarding gates are identified by a letter
- Number of pieces per box are multiple of 100

# General Rules

Rules applicable to any type of data and BD

Three rules for data integrity:

- **Domain Integrity**
- **Entity Integrity**
- **Reference Integrity**

## 2.1 Domain Integrity

# Domain Integrity

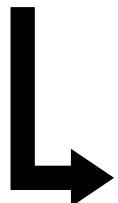
Every value of an attribute must belong to the domain on which we defined it.

Ex: Data type (integer, character or decimal)

Allowed length of data

Null support

When is not possible validate the domain integrity



Application-level checking

# 2.3 Entity Integrity

2.3.1 Motivation

2.3.2 Primary Key

2.3.3 Entities integrity

2.3.4 SQL Syntax

## 2.3.1 Motivation

The tuples (Rows) in a table represent objects / individuals in the real world

The real-world objects are identifiable and distinguishable from others (uniqueness)

As a representant of a real object each tuple is unique (although some attribute values match those of another tuple)

## 2.3.2 Primary Key (PK)

**Candidate Keys** (CK): Minimum set (there are no redundancies) of attributes that uniquely identify each instance

**Primary key** (PK): CK chosen by the designer

## 2.3.2 Primary Key: Example1

Copy
ISBN
Copy #
Barcode
MaxDuration
Penalisation

REQUIREMENTS: The ISBN is a code that is assigned according to the title, author and book publisher.

The barcode is a library internal encoding that is assigned to each copy

Identify CKs and PK

## 2.3.3 Entity Integrity

No component of the primary key can be null

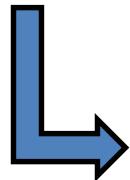
$\text{NULL} \equiv \text{Missing Information}$

NULL is equivalent to an existint object without identification or with an unknown identification

## 2.3.3 Entity Integrity: Remarks

DBMS checks integrity when the primary key is defined

If you do not specify primary key, some DBMS assign PK automatically. Enter an extra attribute to enumerate the tuples.



Automatic numbering is not recommended:

It depends on the insertion order (Independence)

It has no semantic meaning (Domain)

Enter redundancies and dependencies between data

## 2.3.4 SQL Sintax

```
CREATE TABLE <nameT> (
    <nameA1> <domain1> <values>,
    ....
    <nameAk1><domaink1> NOT NULL,
    <nameAk2><domaink2> NOT NULL,
    ....
    <nameAN> <domainN> <values>,
CONSTRAINT nameT_PK PRIMARY KEY (nameAk1,nameAk2)
);
```

## 2.3.4 Syntax SQL: Example1

Simple PK

```
CREATE TABLE PLACE(  
    code NUMBER NOT NULL,  
    name VARCHAR2 (50)  
    Address VARCHAR2 (50)  
    Telephone NUMBER  
    Timetable VARCHAR2 (50)
```

**Constraint PLACE\_PK PRIMARY KEY ( code )**

## 2.3.4 Syntax SQL: Exemple2

### Compound PK

```
CREATE TABLE PRICE_SHOW (
    Code_show NUMBER NOT NULL,
    Code_enclosure NUMBER NOT NULL,
    Zone VARCHAR2 (20) NOT NULL,
    Price NUMBER,
    Constraint Price_Show_PK
    PRIMARY KEY ( Code_show, Code_enclosure_zone )
```

# 2.4 Reference Integrity

2.4.1 Motivation

2.4.2 Foreign Key (FK)

2.4.3 Reference Integrity

2.4.4 SQL Syntax

2.4.5 Referential Diagram

## 2.4.1 Motivation

Relational databases are a set of referenced tables

Student

NIA	Name	DNI	Tel
1	Pepe	46956514	657545454
2	Joan	56056512	666666666

Subject

PK Sub	Name	Classroom
100	BD	1011
202	IA	1013
305	SO	2010



PK St	PK Sub
1	100
2	100
2	202
2	305

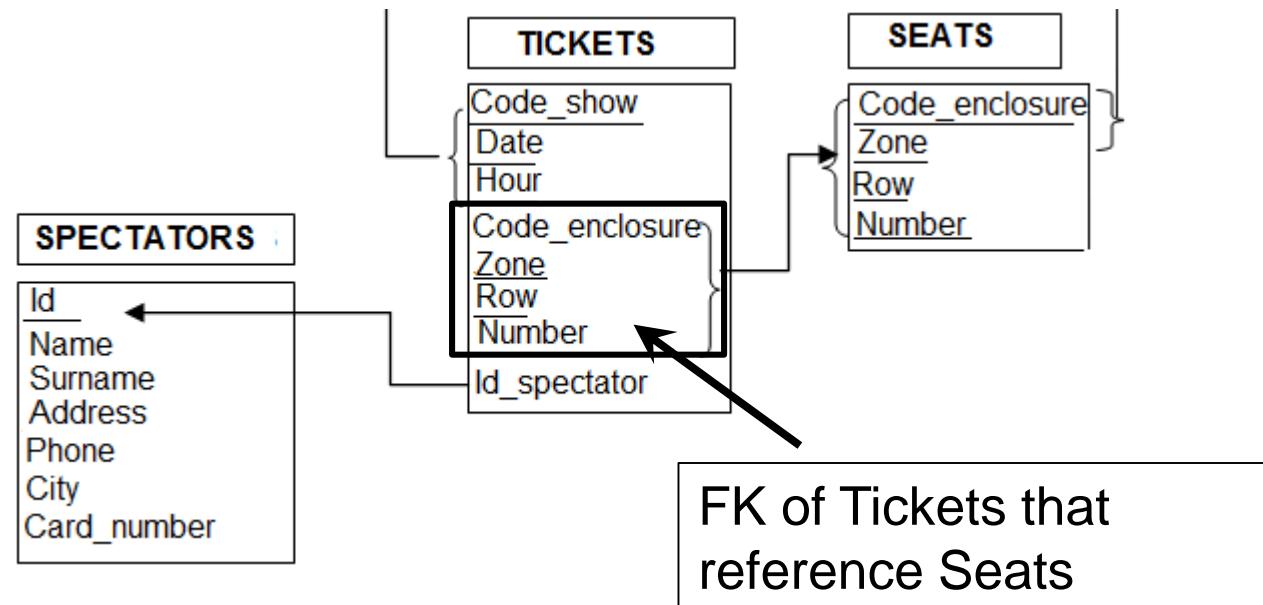


Enrollment

References to tables containing data

## 2.4.2 Foreign Key (FK)

Set of attributes of a relation R2 whose values are the same values as another PK takes in a relationship which refers to R1



OBS: PK and FK must be of the same domain

## 2.4.2 FK: Concepts

Referenced relation. Table containing PK

Referenced tuple. Tuple where exists PK value of FK

Student

<u>NIA</u>	Name	DNI	Tel
1	Pepe	46956514	657545454
2	Joan	56056512	666666666

Subject

<u>PK Sub</u>	Name	Classroom
100	BD	1011
202	IA	1013
305	SO	2010



Enrolment

<u>PK St</u>	<u>PK Sub</u>
2	100
1	100
2	202
2	305

Referential  
Relationship.

Value containing the FK

## 2.4.3 Reference Integrity: Basic Concepts

1. There may be primary key value of R1 that are not referenced by any foreign key of R2

Examples:

- There can be subjects with no students enrolled
- Students can be NOT enrolled

## 2.4.3 Reference Integrity: Basic Concepts

2. A value of FK in a R2 relation referencing to a R1 relation is only allowed if this value appears in the primary key of one of the tuples of R1

Example:

- It makes no sense that a student enrols for a subject that does not exist.
- We can not sell tickets for a flight to the Moon

## 2.4.3 Reference Integrity: Definition

The DB can not have values (NO\_NULL) for foreign key non-primary key of a tuple of the referenced relation

If R2 refers to R1  $\rightarrow$  R1 must exist

The DBMS makes checking whether foreign keys are specified

## 2.4.3 SQL Syntax

**Constraint <nameFK> FOREIGN KEY (<AttributseFK>) REFERENCES <nameReferencedTable> (<AttributesPKReferencedTable>)**

NULL [NOT] ALLOWED

DELETE OF < nameReferencedTable > <effect>

UPDATE OF < AttributesPKReferencedTable > < effect >

Where <effect> can be

RESTRICTED |

CASCADES |

NULLIFIES

## 2.4.3 SQL Syntax: Example

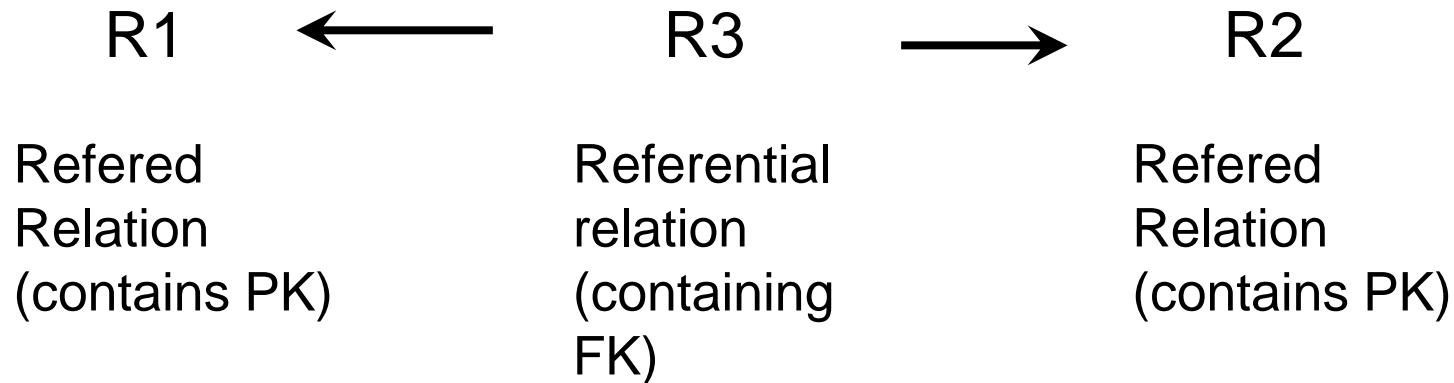
FKs are defined once all tables have been created using the following syntax

ALTER TABLE <nameTable> ADD Constraint

```
ALTER TABLE AUXILIAR ADD CONSTRAINT  
AUXILIAR_PERSONAL_FK FOREIGN KEY ( SS ) REFERENCES  
PERSONAL ( SS ) ;
```

```
ALTER TABLE TICKETS ADD CONSTRAINT TICKETS_TRIP_FK  
FOREIGN KEY ( Code_ticket, Passenger ) REFERENCES  
TRIP( Code_ticket, NIF_Passenger ) ;
```

## 2.4.4 Referential Diagram



Relations can be labeled with foreign keys:



## 2.4.4 Referential Diagram

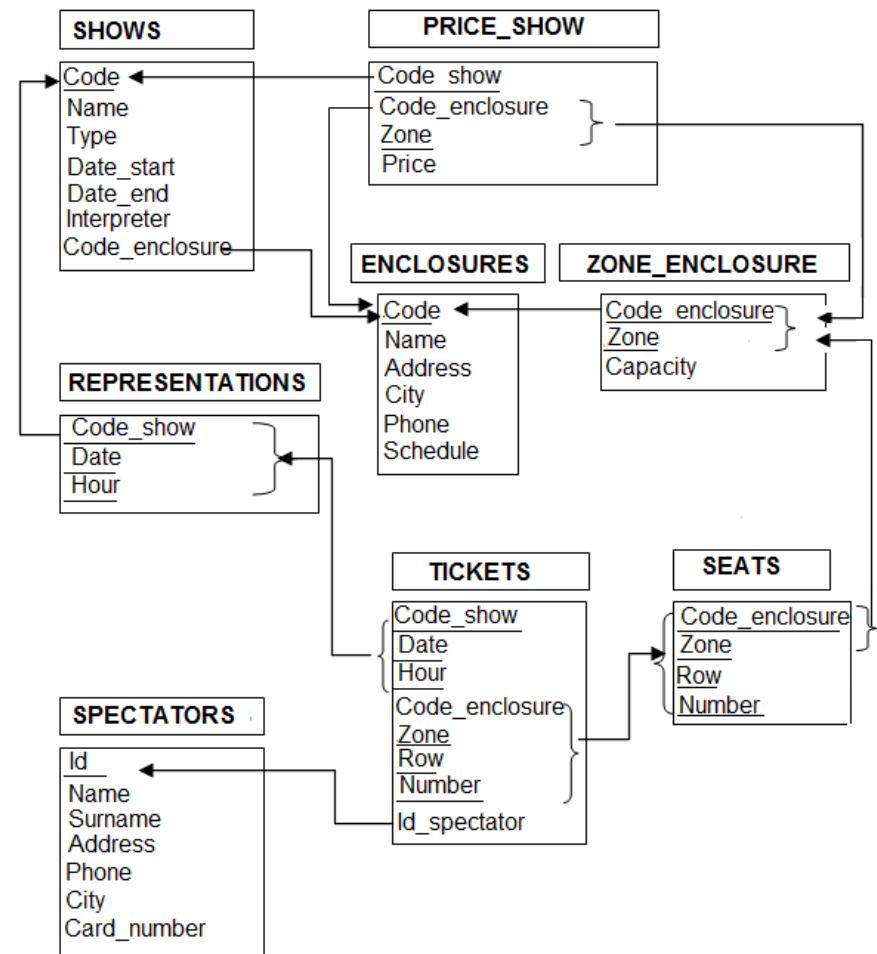
A relation can be referenced and referential

$$R3 \xrightarrow{b} R2 \xrightarrow{a} R1$$

Referential Path:

Referential chain constraints

Determines the order to follow when it modifies (Inserts, Deletes) DB



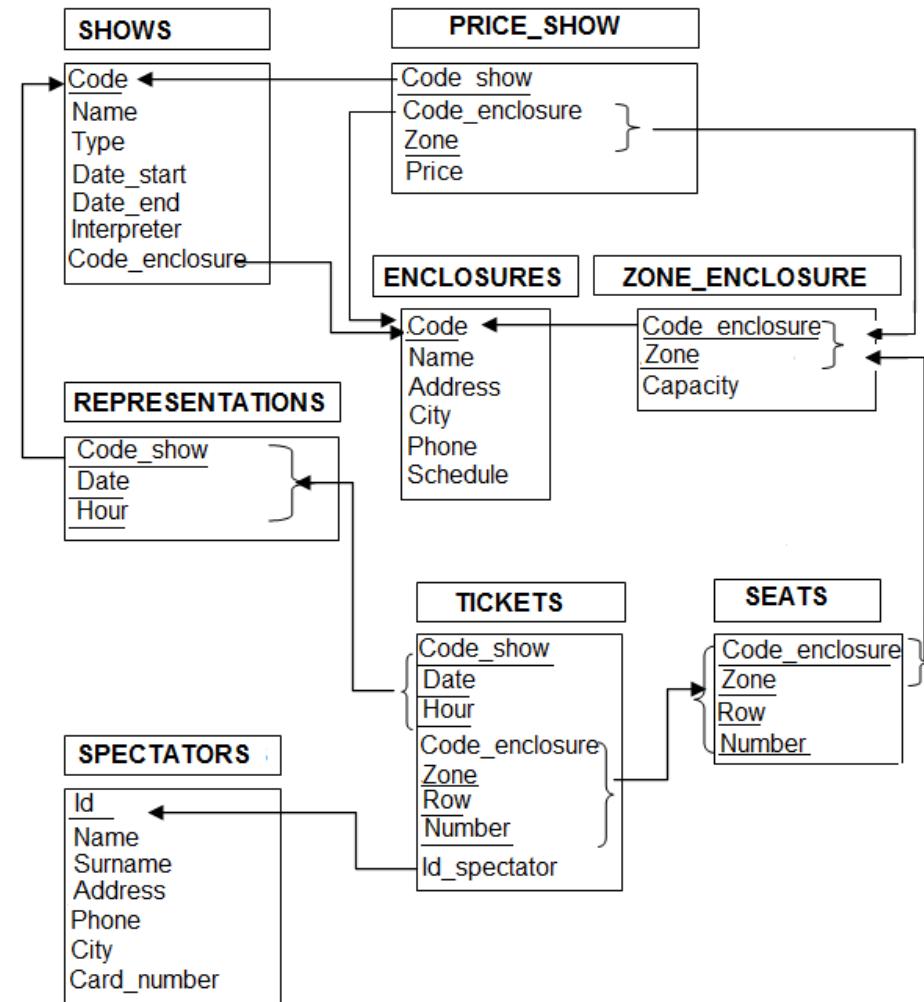
$$R(n) \longrightarrow R(n-1) \longrightarrow \dots \longrightarrow R2 \longrightarrow R1$$

## 2.4.4 Referential diagram:

### Example

Referential Diagram and  
insert / delete order:

1. A enclosure
2. A zone enclosure
3. A seat



# 3. Propagation Constraints

3.1 Motivation

3.2 Null allowed (Nullify)

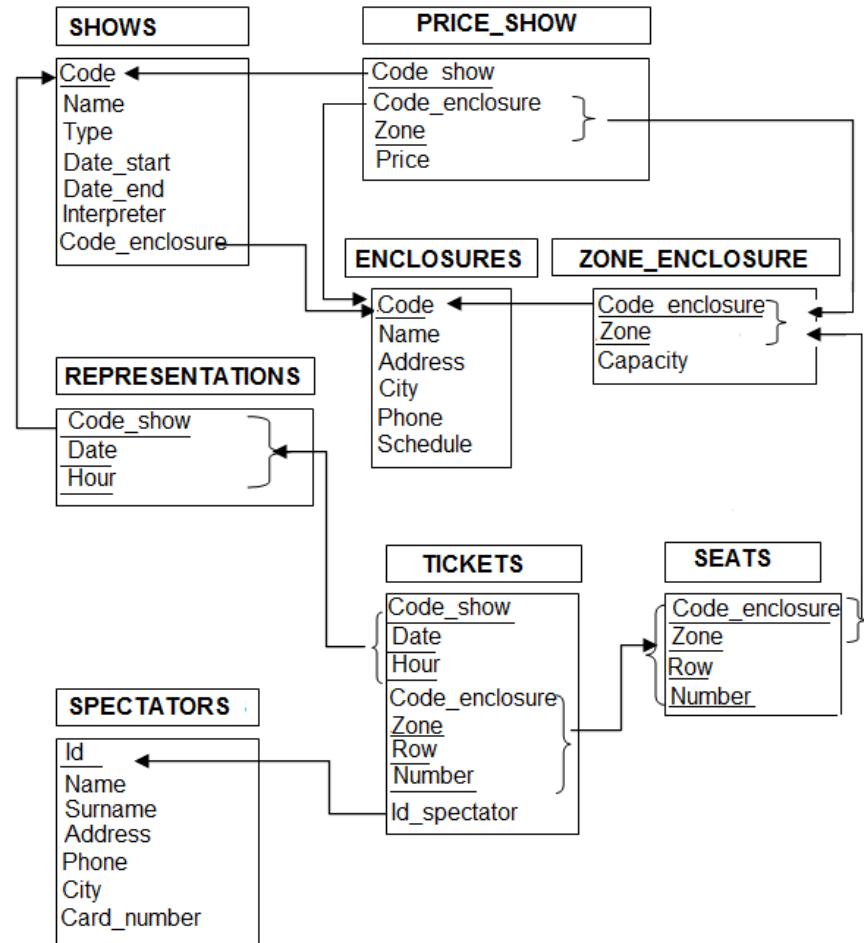
3.3 Delete

3.4 Updates (UpDate)

3.5 Examples

# 3.1 Motivation

What happens if we delete  
a referenced tuple?



## 3.1 Motivation

Foreign keys determine the link between tables

Modifying a referenced tuple affects ALL those referring to it

# 3.1 Motivation

## Rules for foreign keys to guarantee reference integrity

- NULLS acceptation
- Tuples deletion (Delete)
- PK modification (Update)



Restrict (Restricted)  
Propagate (Cascade)  
Accept nulls (Nullifies)

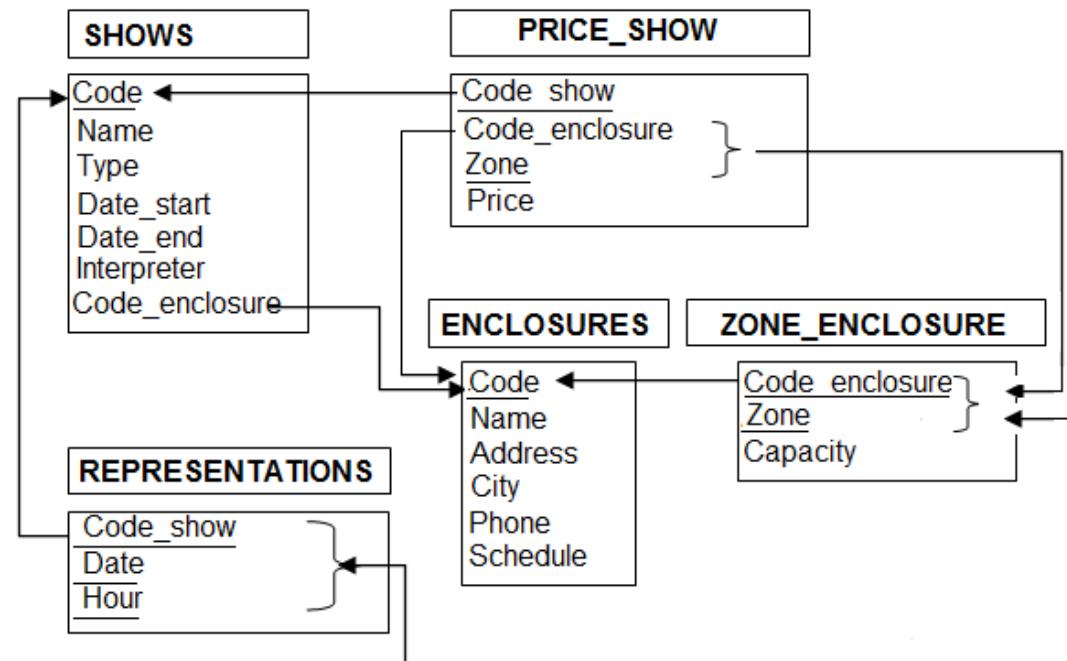
## 3.2 NULLS Acceptation

# NULLS Acceptation

Admitting NULL in a FK not always make sense. It depends on the meaning of the BD and the data policy to be modelled

SQL: By default FK supports NULLs

A show without a venue (price\_show) could make sense in an event organization company.

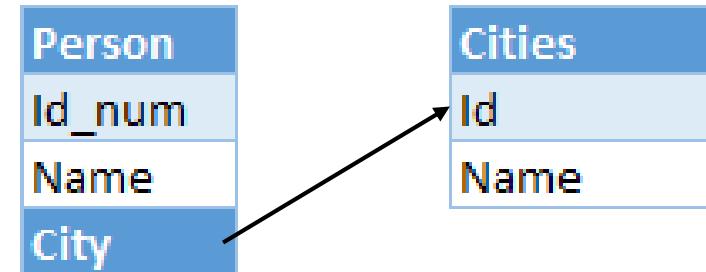


# NULL values meaning

FK can accept NULL values

→ What does it mean?

Id_num	Name	City
234234234	Lucia	002
453454534	Paul	003 NULL
585765837	Ernest	001



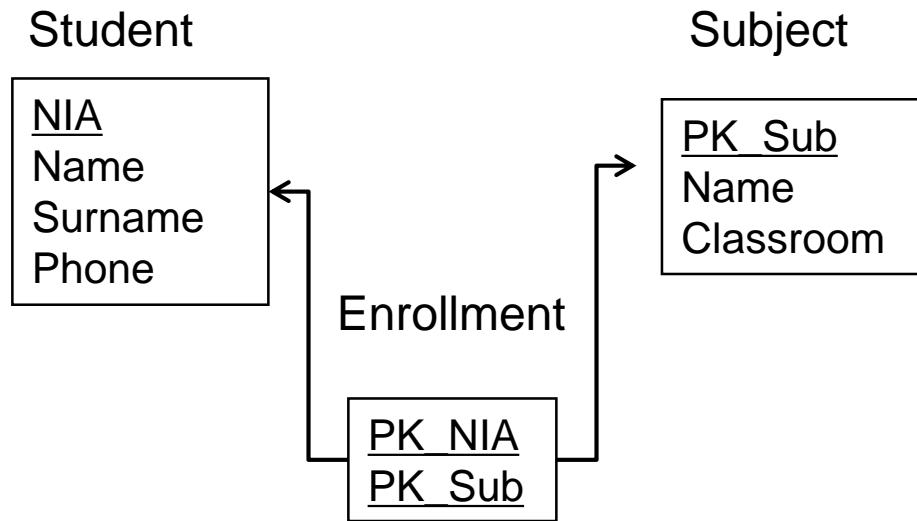
Id	Name
001	Nápoles
002	Barcelona
003	Paris

If we delete a city which is referenced in Person table.  
Their value is replace by NULL.

If FK consists of more than one attribute, they must be entirely NULLs or NOT NULLs

# NULL values meaning

A FK may be part of the PK of the referential relation  
(should it?)



In this case the Enrollment table must not allow Null values in their FKS

## 3.3 Delation (Delete)

3.3.1 Restricted

3.3.2 Cascade

3.3.3 Nullify

### 3.3.1 Restricted

- Delete S2 supplier from S Relation

→ **Not possible**

S	S#	Snom	Zona	Ciutat
	S1	Salazar	20	Londres
	S2	Jaime	10	Paris
	S3	Bernal	30	Paris
	S4	Corona	20	Londres
	S5	Aldana	30	Atenas

SP	S#	P#	Cant
	S1	P1	300
	S1	P2	200
	S2	P1	300
	S2	P2	400
	S3	P3	200
	S4	P2	200

P	P#	Pnom	Color	Pes	Ciutat
	P1	Femella	Vermell	12	Londres
	P2	Pern	Verd	17	Paris
	P3	coixinet	Blau	17	Paris

### 3.3.2 Cascade

- Delete S2 supplier from S Relation

Deleted in  
S, SP

S	S#	Snom	Zona	Ciutat
	S1	Salazar	20	Londres
	S2	Jaime	10	Paris
	S3	Bernal	30	Paris
	S4	Corona	20	Londres
	S5	Aldana	30	Atenas

SP	S#	P#	Cant
	S1	P1	300
	S1	P2	200
	S2	P1	300
	S2	P2	400
	S3	P3	200
	S4	P2	200

P	P#	Pnom	Color	Pes	Ciutat
	P1	Femella	Vermell	12	Londres
	P2	Pern	Verd	17	Paris
	P3	coixinet	Blau	17	Paris

### 3.3.2 Cascade

- Delete S2 supplier from S Relation

Deleted in  
S, SP

S	S#	Snom	Zona	Ciutat
	S1	Salazar	20	Londres
	S2	Jaime	10	Paris
	S3	Bernal	30	Paris
	S4	Corona	20	Londres
	S5	Aldana	30	Atenas

SP	S#	P#	Cant
	S1	P1	300
	S1	P2	200
	S3	P3	200
	S4	P2	200

P	P#	Pnom	Color	Pes	Ciutat
	P1	Femella	Vermell	12	Londres
	P2	Pern	Verd	17	Paris
	P3	coixinet	Blau	17	Paris

### 3.3.2 Cascade

- Delete S2 supplier from S Relation

Deleted in  
S, SP

S	S#	Snom	Zona	Ciutat
	S1	Salazar	20	Londres
	S3	Bernal	30	Paris
	S4	Corona	20	Londres
	S5	Aldana	30	Atenas

SP	S#	P#	Cant
	S1	P1	300
	S1	P2	200
	S3	P3	200
	S4	P2	200

P	P#	Pnom	Color	Pes	Ciutat
	P1	Femella	Vermell	12	Londres
	P2	Pern	Verd	17	Paris
	P3	coixinet	Blau	17	Paris

### 3.3.3 Nullifies

- Delete S2 supplier from S Relation

S	S#	Snom	Zona	Ciutat
	S1	Salazar	20	Londres
	S2	Jaime	10	Paris
	S3	Bernal	30	Paris
	S4	Corona	20	Londres
	S5	Aldana	30	Atenas

→ **NULL values  
in SP**

SP	S#	P#	Cant
	S1	P1	300
	S1	P2	200
	S2	P1	300
	S2	P2	400
	S3	P3	200
	S4	P2	200

P	P#	Pnom	Color	Pes	Ciutat
	P1	Femella	Vermell	12	Londres
	P2	Pern	Verd	17	Paris
	P3	coixinet	Blau	17	Paris



### 3.3.3 Nullifies

- Delete S2 supplier from S Relation

S	S#	Snom	Zona	Ciutat
	S1	Salazar	20	Londres
	S2	Jaime	10	Paris
	S3	Bernal	30	Paris
	S4	Corona	20	Londres
	S5	Aldana	30	Atenas

→ **NULL values  
in SP**

SP	S#	P#	Cant
	S1	P1	300
	S1	P2	200
	NULL	P1	300
	NULL	P2	400
	S3	P3	200
	S4	P2	200

P	P#	Pnom	Color	Pes	Ciutat
	P1	Femella	Vermell	12	Londres
	P2	Pern	Verd	17	Paris
	P3	coixinet	Blau	17	Paris



### 3.3.3 Nullifies

- Delete S2 supplier from S Relation

S	S#	Snom	Zona	Ciutat
	S1	Salazar	20	Londres
	S3	Bernal	30	Paris
	S4	Corona	20	Londres
	S5	Aldana	30	Atenas

→ **NULL values  
in SP**

SP	S#	P#	Cant
	S1	P1	300
	S1	P2	200
	NULL	P1	300
	NULL	P2	400
	S3	P3	200
	S4	P2	200

P	P#	Pnom	Color	Pes	Ciutat
	P1	Femella	Vermell	12	Londres
	P2	Pern	Verd	17	Paris
	P3	coixinet	Blau	17	Paris

## 3.4 UpDate

3.4.1 Restricted

3.4.2 Nullify

3.4.3 Cascade

## 3.4.1 Restricted

- Update S2 value from S Relation

Not possible

S	S#	Snom	Zona	Ciutat
	S1	Salazar	20	Londres
	S2	Jaime	10	Paris
	S3	Bernal	30	Paris
	S4	Corona	20	Londres
	S5	Aldana	30	Atenas

SP	S#	P#	Cant
	S1	P1	300
	S1	P2	200
	S2	P1	300
	S2	P2	400
	S3	P3	200
	S4	P2	200

P	P#	Pnom	Color	Pes	Ciutat
	P1	Femella	Vermell	12	Londres
	P2	Pern	Verd	17	Paris
	P3	coixinet	Blau	17	Paris

## 3.4.2 Cascade

- Update P2 value from P Relation

we change the values in P, SP

S	S#	Snom	Zona	Ciutat
S1	Salazar	20	Londres	
S2	Jaime	10	Paris	
S3	Bernal	30	Paris	
S4	Corona	20	Londres	
S5	Aldana	30	Atenas	

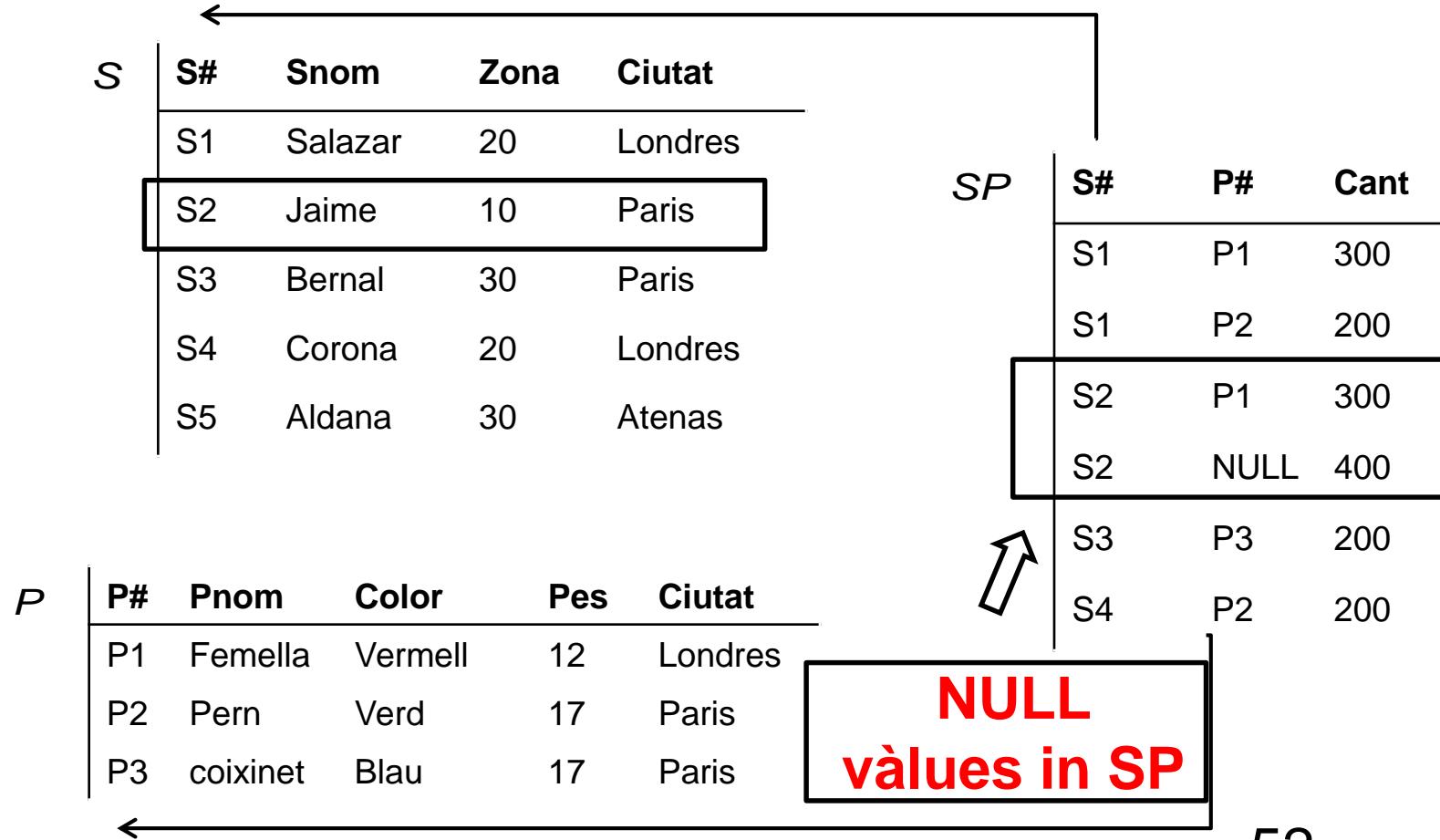
SP	S#	P#	Cant
S1	P1	300	
S1	P2	200	
S2	P1	300	
S2	P2	400	
S3	P3	200	
S4	P2	200	

P	P#	Pnom	Color	Pes	Ciutat
P1	Femella	Vermell	12	Londres	
P2	Pern	Verd	17	Paris	
P3	coixinet	Blau	17	Paris	

### 3.4.3 Nullifies

In what order should it be done?

- Update P2 by P5 value from P Relation



# Observations

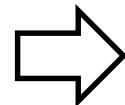
- CASCades foreign key rule can lead to problems



Reference: CASCades

What if we delete tuples at R1?

If the rule a, b does not  
allow to delete tuples in R3  
as a result of removing  
elements in R2

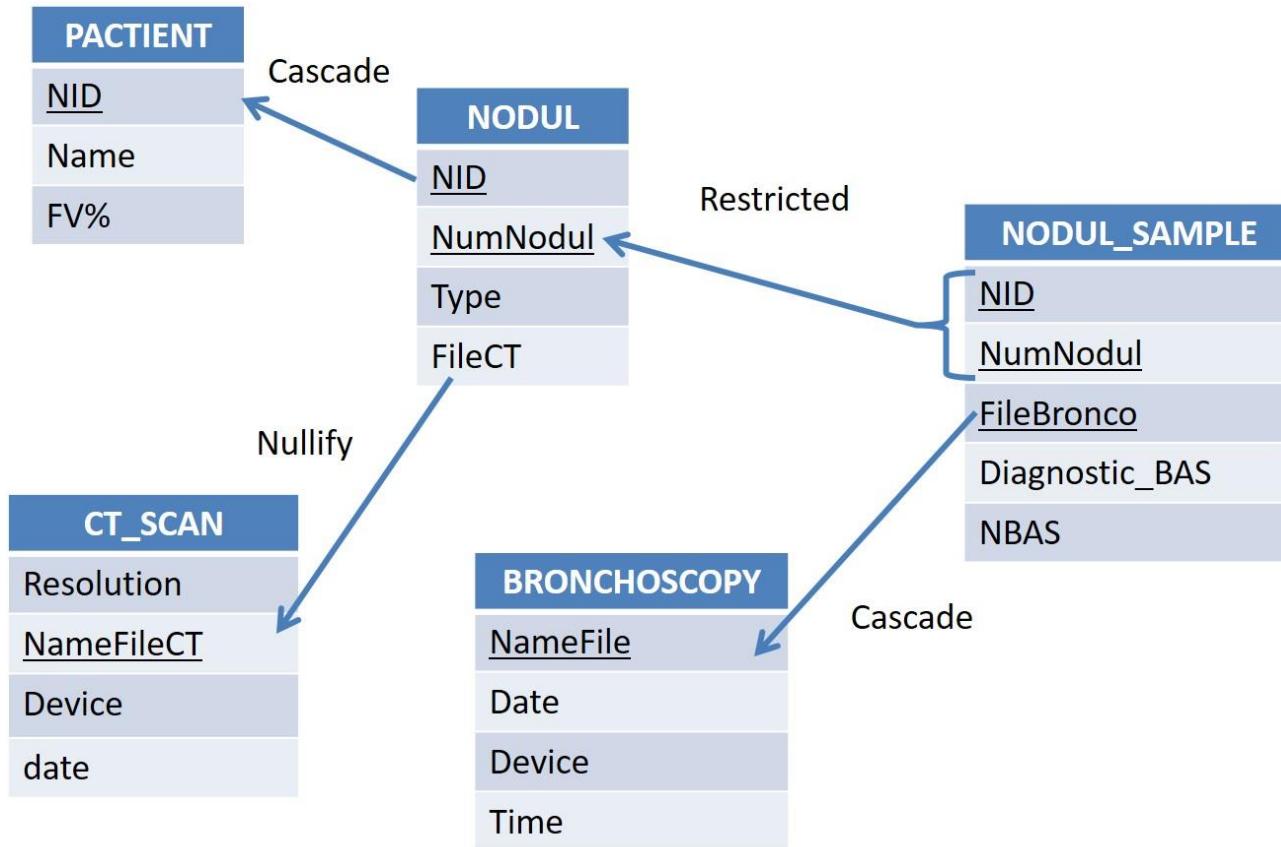


R1 tuple **can not** be  
deleted

# Observations

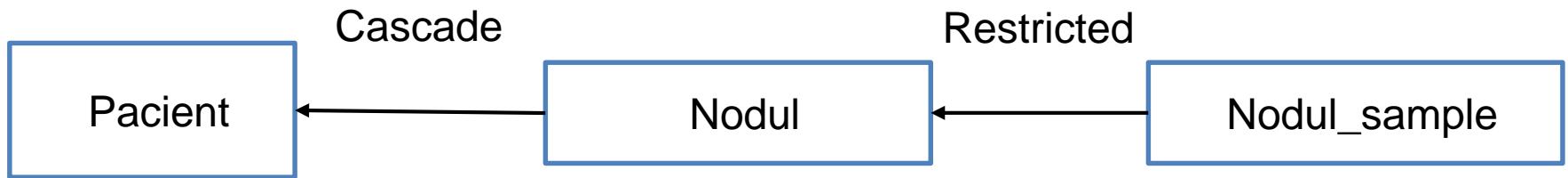
Restricted (default mode) is the rule that has fewer risks and it is easier to implement but less comfortable for the user (deleting with a certain order)

# Example



## Referential diagram

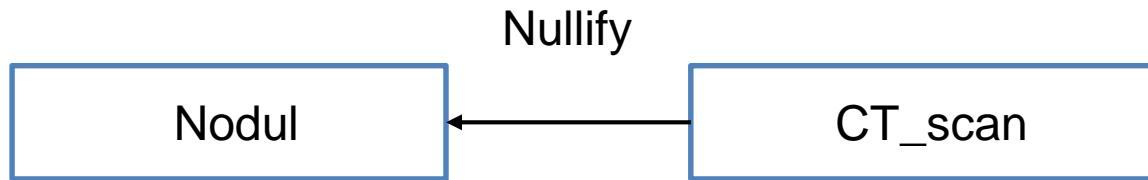
What happens if we delete a patient?



- 1) *A patient (NID) could be automatically removed if It doesn't have related nodules.*
- 2) *If patient has nodules:*
  - *If these nodules doesn't have related Nodul\_sample, these nodules will be automatically removed too (cascade)*
  - *If these related nodules have Nodul\_sample they have to be removed first manually. (Restricted)*  
*The order to delete will be : Nodul\_sample  
Pacient*

## What happens if we delete a CT\_scan?

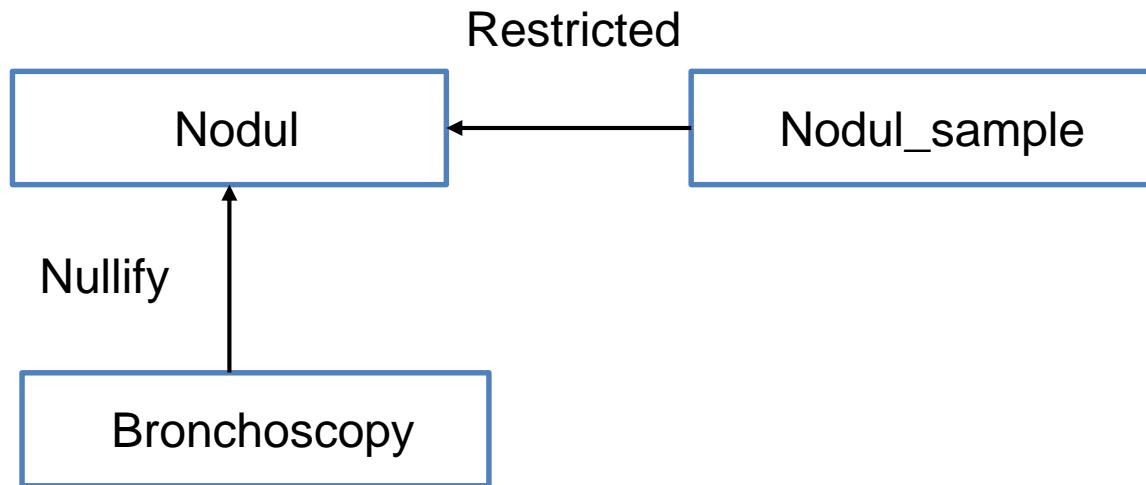
### Referential diagram



*The value of the field File\_CT will be updated as Null in all records which contains the deleted CT.*

## Referential diagram

What happens if we delete a nodul?



- 1) *A nodul could be automatically deleted if Nodul\_sample doesn't have any reference of it.*
- 2) *If the Nodul\_sample has the nodul, first we need to remove from it and then from Nodul table because the restricted ruler.*