

MySQL

Laia Subirats Maté

11 de maig de 2021

Índex de continguts

1 Disseny de bases de dades

1. Models conceptuals de bases de dades
2. El model Entitat-Relació
3. El model Entitat-Relació estès

2 El model relacional

1. Introducció al model relacional
2. Claus en el model relacional
3. Operacions bàsiques
4. Teoria de la normalització

3 Definició i manipulació de dades

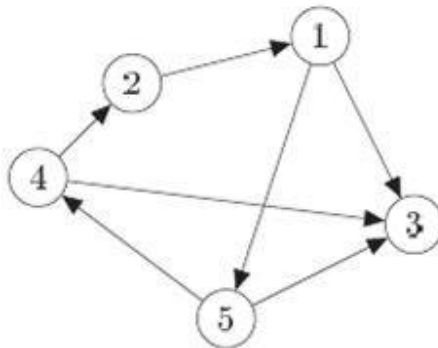
1. Llenguatge de definició de dades, DDL
2. Llenguatge de manipulació de dades, DML

1 Disseny de bases de dades

Un cop feta l'anàlisi del projecte, s'inicia el disseny de la base de dades, que consisteix en l'elaboració d'un diagrama, concretament, d'un model entitat relació (ER), amb tota la seva documentació. Necessitem conceptes de la teoria de grafs.

Ens fixem, especialment, amb els **grafs dirigits**, que són conjunts de **nodes** i **arcs**, que uneixen parelles de nodes de forma dirigida (mitjançant fletxes).

- Un **camí** és una seqüència de nodes consecutius, units mitjançant arcs. Un **cicle** és un camí que comença i acaba en el mateix node.
- El **grau** d'un node és el seu nombre de veïns. Un **terminal** és un node de grau 1.
- Un **graf connex** és el que té un camí per qualsevol parella de nodes.



1 Models conceptuais de bases de dades

Els **models de dades** s'acostumen a presentar com a grafs. Cada node simbolitza un concepte de l'univers de l'aplicació. Per exemple, un diagrama de classes o un model de domini.

El **model de domini** és el més abstracte dels models que constitueixen l'especificació tècnica d'una aplicació. Conté les nocions essencials del projecte.

Les màximes de qualitat d'un disseny són:

- Prohibir la **redundància**, no guardar una mateixa informació en llocs diferents
- **Unificar conceptes**, conceptes allunyats d'un model de domini no han de tenir el mateix nom.

1.1 El model ER

Les **entitats** són conjunts d'elements dels quals ens interessa mantenir una mateixa agrupació heterogènia de dades. Són nodes del graf que és el model ER (estructures en C, Pascal o classes en Java)

És el concepte més bàsic de disseny, cada entitat té un nom (SUBSTANTIU SINGULAR EN MAJÚSCULES) i la seva representació gràfica és un rectangle.



PERSONA

1.1 El model ER

- Els **atributs** (o camps) són cadascuna de les dades que convingui agregar a una entitat. Convé esforçar-se en que el nom dels atributs consti d'una sola paraula, és important el concepte d'abstracció, calen continguts semàntics i no "tipus".
- Normalment seran substantius en singular per les cadenes de caràcters, en plural pels enters i adjectius pels booleans, sempre en minúscula. Es representa gràficament amb una el·lipse que l'uneix a l'entitat que es estructura.



1.1 El model ER

Les **claus** identifiquen els elements d'una entitat. Totes les entitats han de tenir un conjunt d'atributs clau (normalment, d'un sol atribut) que per definició, tindrà una combinació de valors diferents en cada element de l'entitat.

Això garanteix que les entitats siguin conjunts i no multiconjunts, ja que el valor que poden tenir ve condicionat per algun contingut de la base de dades. Per tant, no convé utilitzar l'atribut nom de PERSONA com a atribut clau.



1.1 El model ER

Els atributs **multivalorats** són aquells que ens permeten tenir diferents valors en algun element d'una entitat. Es tracta de col·leccions, no hi ha cap ordre entre els valors d'un atribut multivalorat.

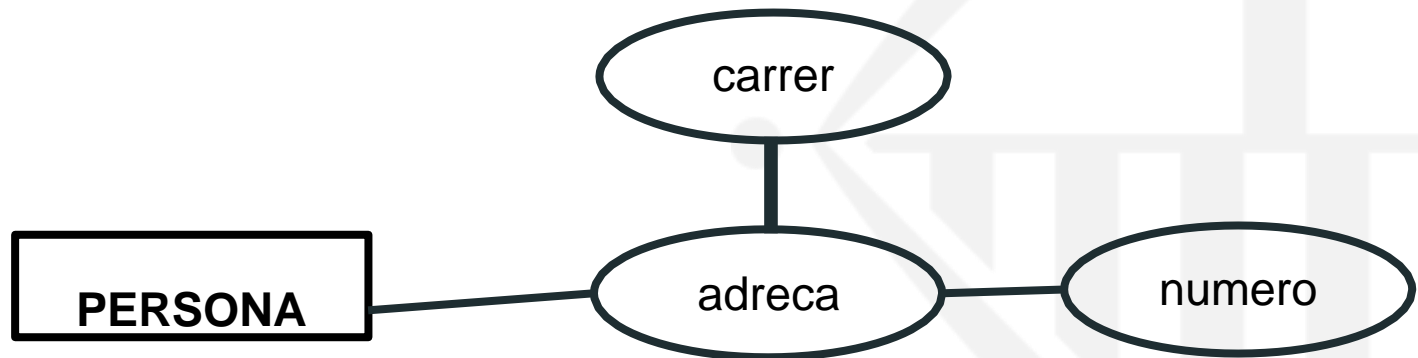
Es fan servir poc, però resulten interessants en l'anàlisi. La qüestió està en si els valors de diferents elements de l'entitat es poden repetir, com en el cas del telèfon.

S'utilitzen substantius en plural i en minúscula.



1.1 El model ER

Els atributs **compostos** són aquells que ens permeten establir una jerarquia entre atributs, és a dir, els atributs compostos contenen atributs més petits. És a dir, ens pot interessar tractar les adreces de les PERSONES però també fer tractaments segons els carrers.



1.1 El model ER

Els atributs **calculats** (o derivats) són aquells els valors dels quals són autogestionats. Normalment serveixen per mantenir informacions estadístiques.

En el moment d'afegir-los al disseny, no cal preocupar-se de qui ni quan n'actualitzarà els seus valors. Es tracta d'una gestió minuciosa del temps de càlcul del valor i, per tant, la deicisió d'usar-los està relacionada amb l'eficència d'aquest càlcul

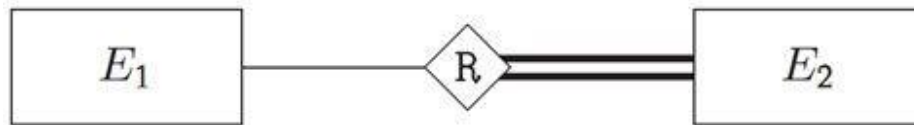


1.1 El model ER

Les **relacions** en un model ER lliguen dues o més entitats. L'estructuració de les bases de dades prové de les relacions entre entitats, per això són tan importants.

El **grau** d'una relació, és el nombre d'entitats que relaciona, binàries o ternàries. La participació d'una entitat en una relació pot ser total o parcial.

En el següent diagrama tenim una relació R que relaciona dues entitats E1 i E2.



1.1 El model ER

- La participació parcial (línia simple) vol dir que pot haver-hi elements d'E1 que no siguin apuntats des de R.
- La participació total (línia doble) vol dir que tots els elements d'E2 han d'estar relacionats forçosament amb algun d'E1.
- Si E1 fos PAIS i E2 fos PERSONA, acceptaríem països encara que no hi hagués cap persona, però no persones si no sapiguéssim de quin país són.



1.1 El model ER

Les participacions total suposen dependència d'existència, els DBMS tenen mecanismes per automatitzar actualitzacions, per tant, difícilment dues entitats poden tenir participació total alhora en la relació que les lliga.

La **cardinalitat** és el nombre d'elements d'una entitat que poden estar associats a un cert element de l'altre entitat de la relació. La noció de cardinalitat reflecteix relacions de pertinença d'elements d'una entitat en elements d'una altra.

Per relacions binàries hi ha tres tipus de cardinalitat 1:1, 1:N i M:N.

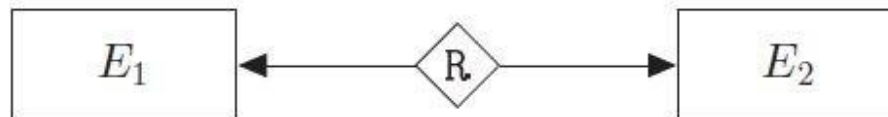
1.1 El model ER

Les **relacions binàries amb cardinalitat 1:1** permeten a cada element d'E1 associar-se amb un sol element d'E2 (o amb cap) i al revés.

Gairebé sempre són relacions supèrflues i poden suprimir-se del diagrama agregant els atributs de les dues entitats en una de sola. En cas de no ser així pot ser:

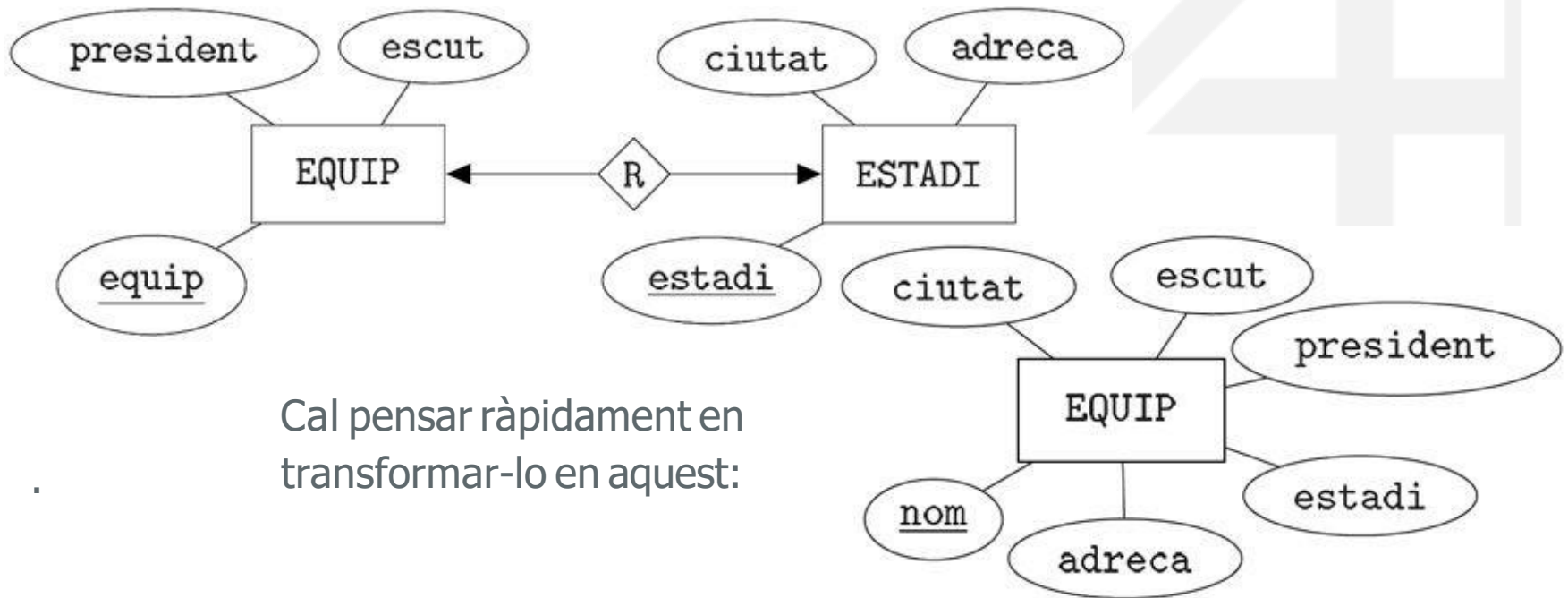
1. Perquè una de les entitats té alts índex d'activitat
2. Perquè totes dues entitats són grans i hi ha molts zeros a algun costat

Les fletxes volen dir 1 o cap.



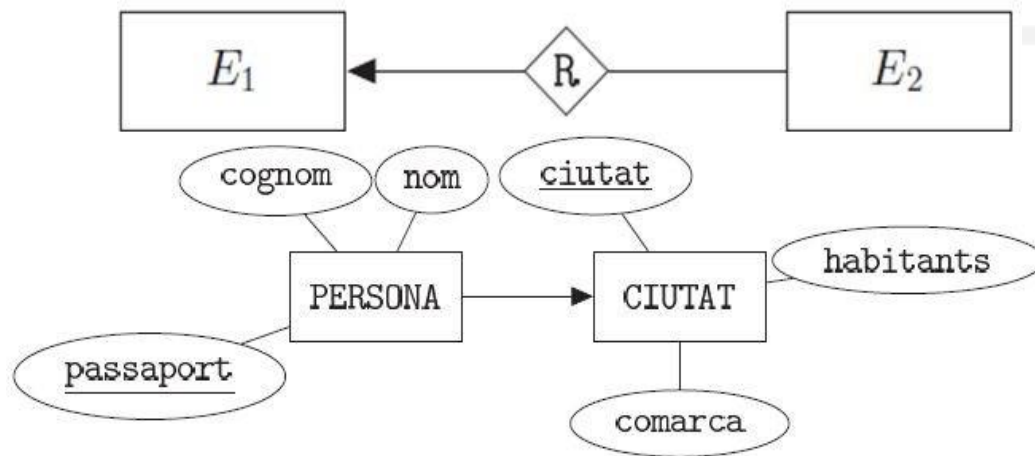
1.1 El model ER

Davant d'un model com aquest:



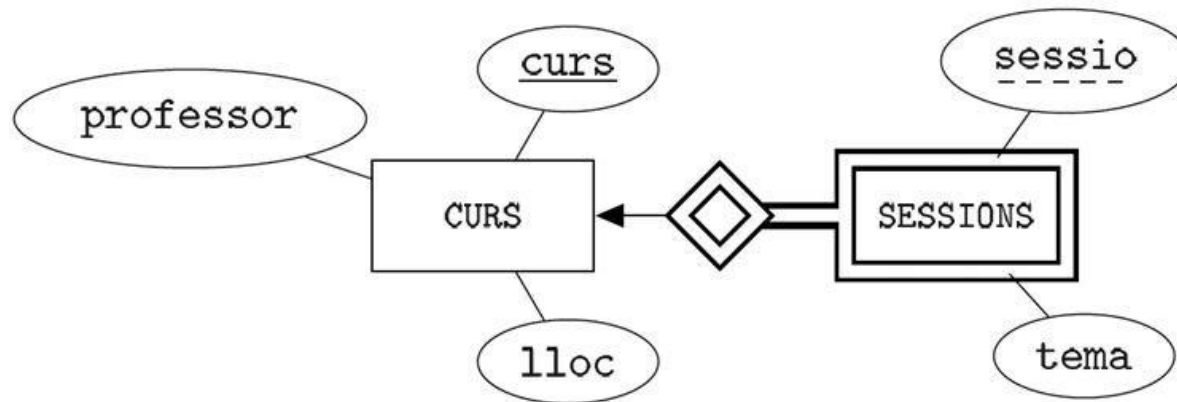
1.1 El model ER

- Les **relacions amb cardinalitat 1:N** permeten a cada element d'E1 associar-se amb diversos elements d'E2 o amb cap, però en canvi, només es permet associar cada element d'E2 amb un únic element d'E1 com a molt. Es pot prescindir del símbol i el nom de la relació.



1.1 El model ER

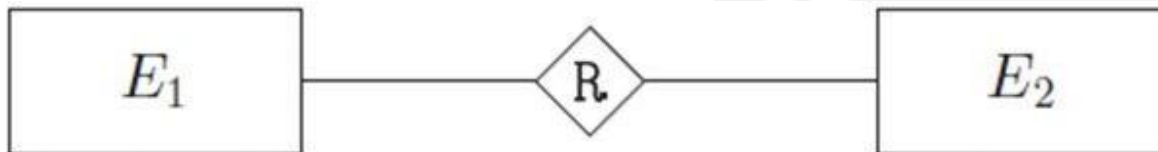
- Les entitats **febles** són les que estan al costat N de la relació, a les del costat 1 se les anomena entitats **fortes**. Per tant, una entitat feble només pot tenir una entitat forta.
- Una entitat feble ha de tenir un atribut que es diu discriminant, per distingir dintre dels elements relacionats amb un mateix element de l'entitat forta. Una entitat feble és una extensió d'un atribut multivalorat. Anomenem les entitats febles amb substantius en plural.



1.1 El model ER

Les relacions binàries amb cardinalitat $M:N$ permeten a cada element d'E1 associar-se amb molts elements d'E2, així com a cada element d'E2 associar-se amb molts d'E1.

Recuperant el producte cartesià, si E1 té n elements i E2 té m elements, R en pot arribar a tenir com a màxim el valor de la multiplicació $n \times m$.

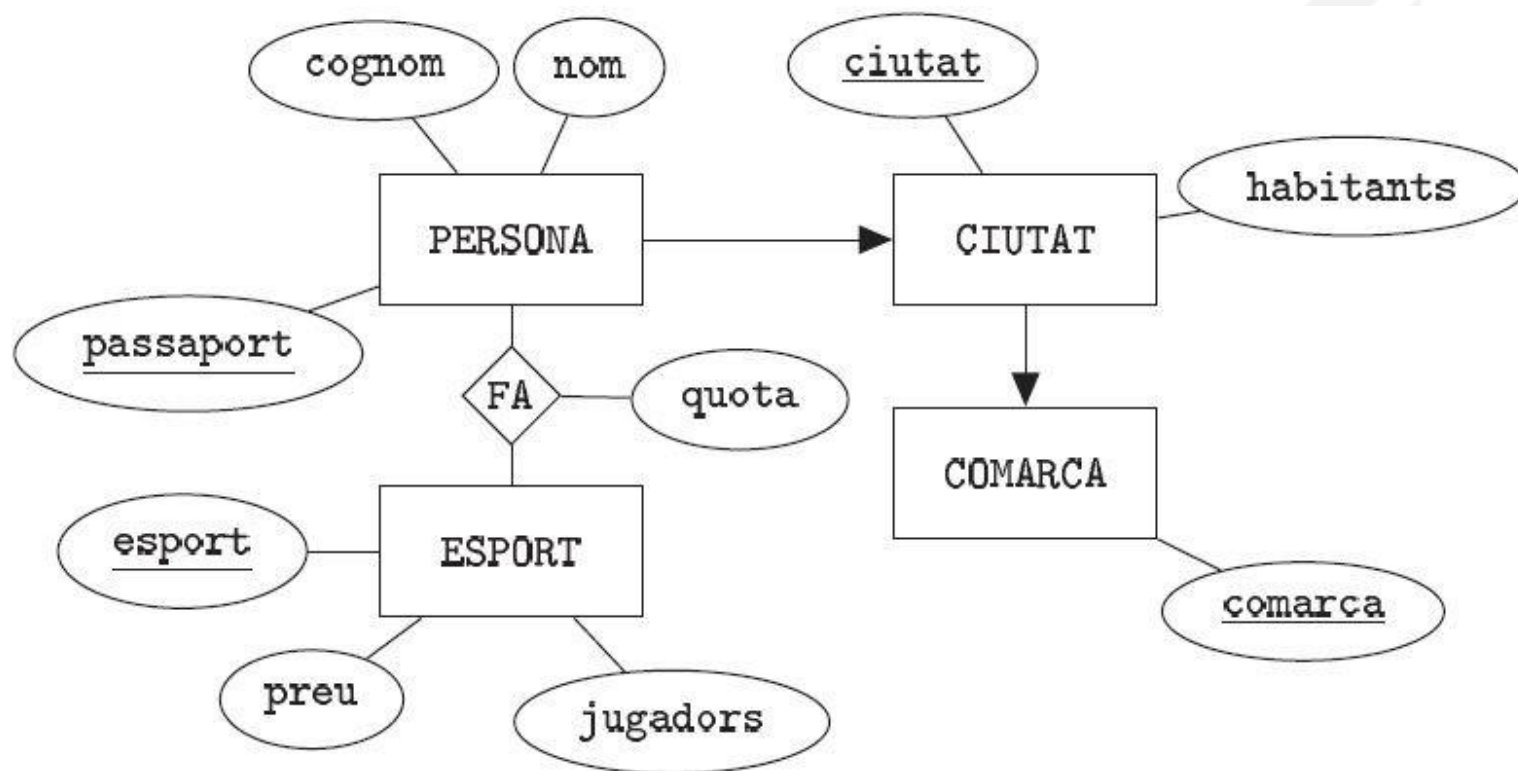


Per disposar d'aquest espai, cal considerar les relacions $M:N$ com a nous nodes en el model ER.

1.1 El model ER

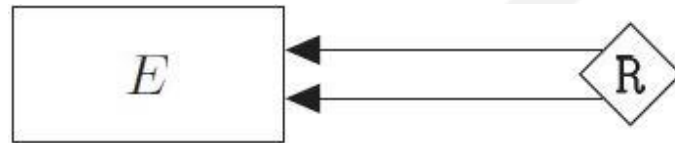
- Les relacions M:N poden tenir atributs, cosa que fa que encara s'assemblin més a les entitats, però no tenen clau, i l'usuari no és capaç d'identificar-ne els elements.
- Per saber si una relació és M:N ens podem preguntar si un E1 pot relacionar-se amb molts E2 i un E2 pot relacionar-se amb molts E1.
- En l'exemple del club esportiu, cal guardar quins esports practica cada soci i quant paga per cadascun i a part, cada esport té el seu preu per defecte i el nombre de jugadors que calen per formar un equip (Una persona pot practicar molts esports i un esport pot ser practicat per moltes persones).

1.1 El model ER

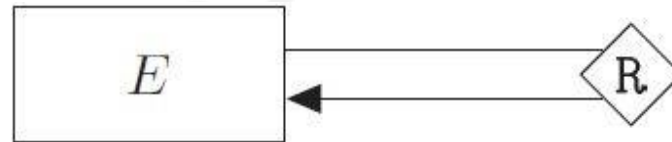


1.1 El model ER

Les autorelacions 1:1 són seqüències, classes d'equivalència. Com les relacions entre dues entitats, no s'utilitzen gaire.



Les autorelacions 1:N permeten expressar jerarquies en forma d'arbre de predecessors. Són més genèriques.



Les autorelacions M:N expressen dependència entre elements d'una mateixa entitat, de manera que un element pot ser relacionat amb molts, i al mateix temps, molts elements es poden relacionar amb l'un.

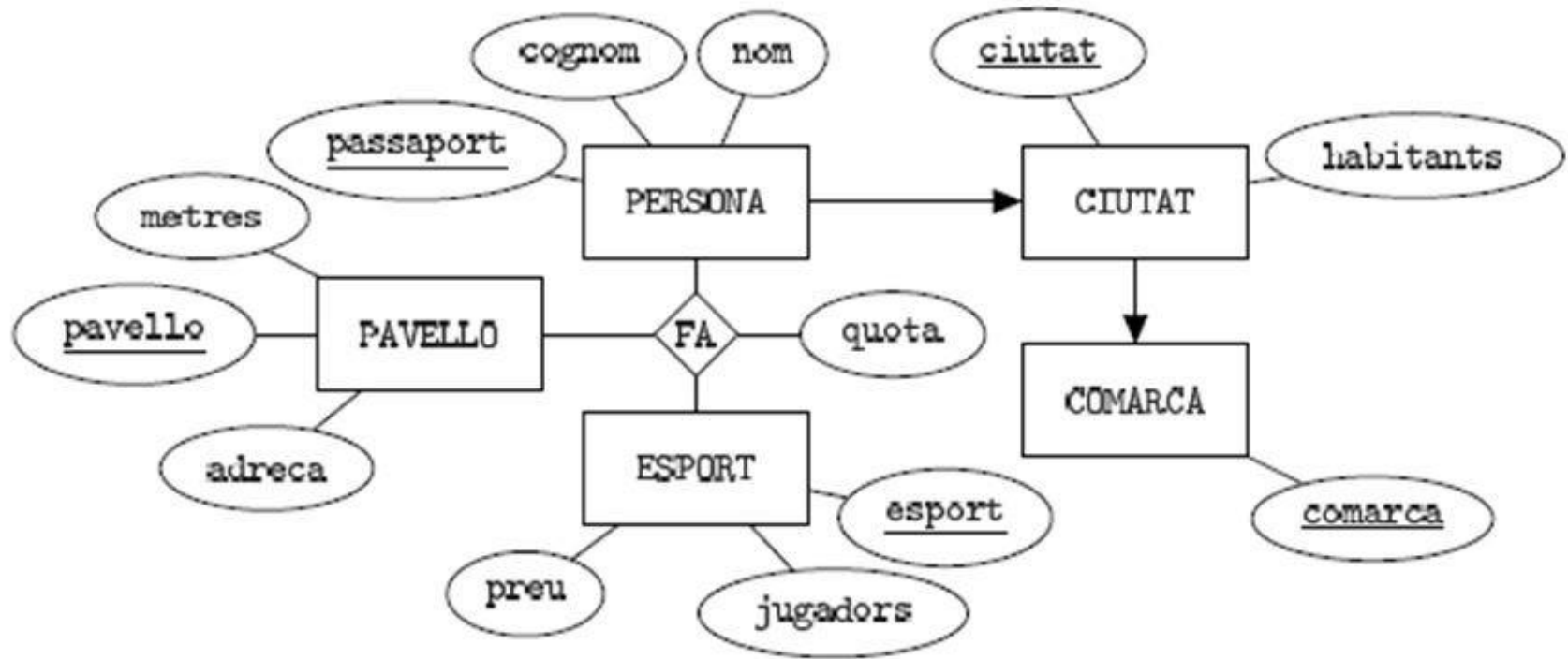
1.1 El model ER

Les relacions ternàries es donen entre 3 entitats i una circumstància que les involucra a totes tres. Representen un nou node en el model ER i el domini de la relació és el conjunt resultant del producte cartesià entre les tres entitats.

Preguntem-nos si qualsevol E1 pot estar relacionat amb qualsevol E2 i qualsevol E3...

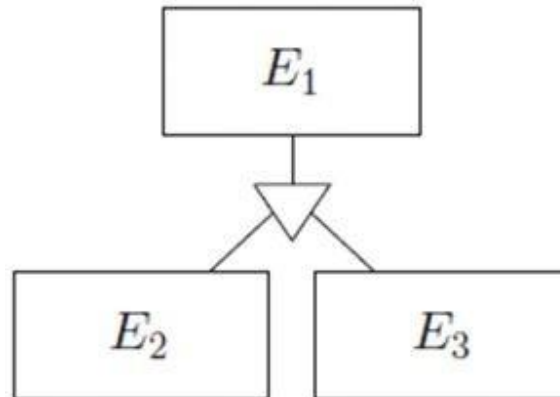
En les relacions ternàries, es creen nodes nous independentment de la cardinalitat, per tant, cal anar alerta amb la creació de nuls estructurals. No hi pot haver relacions ternàries amb dues fletxes.

1.1 El model ER

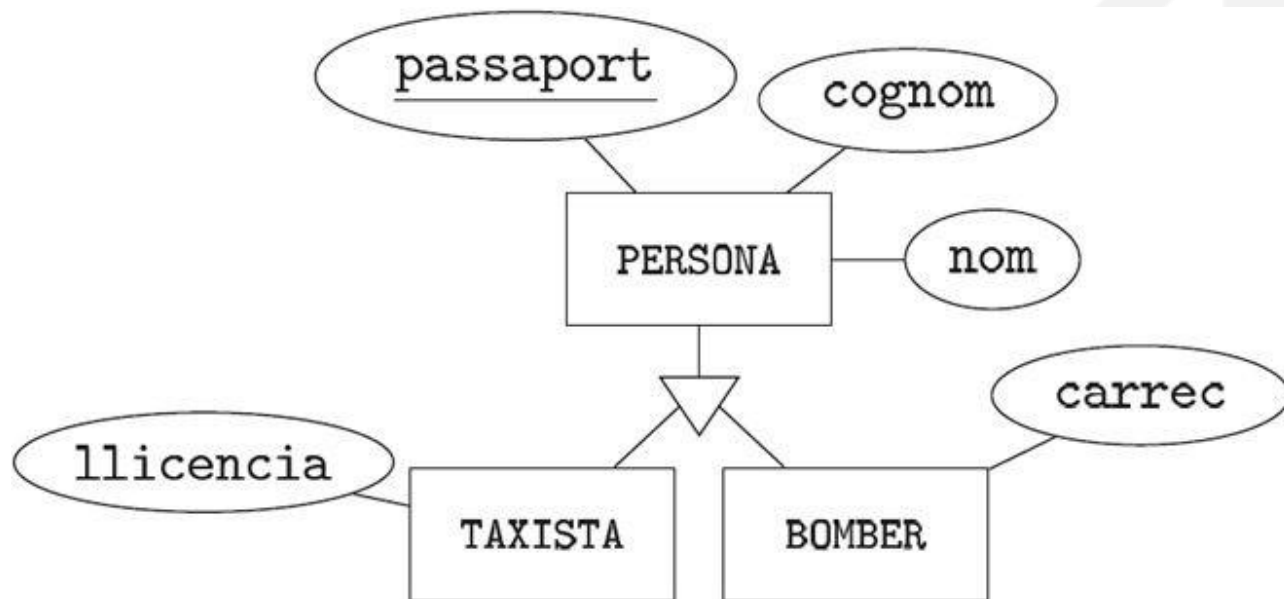


1.2 El model ER estàs

L'especialització i la generalització d'entitats és una aportació de la programació orientada a objectes. E_1 s'anomena entitat genèrica i E_2 i E_3 són les entitats especialitzades.



1.2 El model ER estàs



2. El model relacional

- En el seu sentit més genèric, l'àlgebra estudia els comportaments i el potencial expressiu d'unes estructures en les quals bàsicament hi ha definits uns valors, així com unes operacions internes per treballar amb aquests valors.
- Per exemple, en l'Àlgebra de Boole amb els valors $\{0,1\}$ i les operacions NOT, OR, AND...
- En l'àlgebra relacional, en canvi, els valors, en lloc del zero i l'u, són relacions, cosa que les converteix en el concepte central de la teoria.

2. El model relacional

Un dels aspectes que segurament introdueix més confusió a l'hora d'adquirir la terminologia pròpia de les bases de dades és el canvi semàntic que va sofrint el terme relació al llarg del temps. Fins ara, l'hem utilitzat bàsicament com un arc del graf del model ER, tot i que també, per cardinalitats complexes, s'han vist relacions que generen nodes.

Aquí és diferent. Dins l'àlgebra relacional reprenem el concepte de conjunt vist en el capítol 1 per a les relacions. Les relacions són conjunts, o si voleu llistes, per imaginar una cosa més concreta. I recordeu que ja hem insistit que per definició un conjunt no té elements repetits. És a dir, en l'àlgebra relacional, per definició les relacions no tenen tuples iguals.

2. El model relacional

relació [*s. XIV; del ll. relatio, -ōnis, íd.*]

f **1** 1 Acció de relatar o referir allò que hom ha sentit, ha vist, etc. *Una relació detallada dels fets. Relació oral, escrita. (...)*

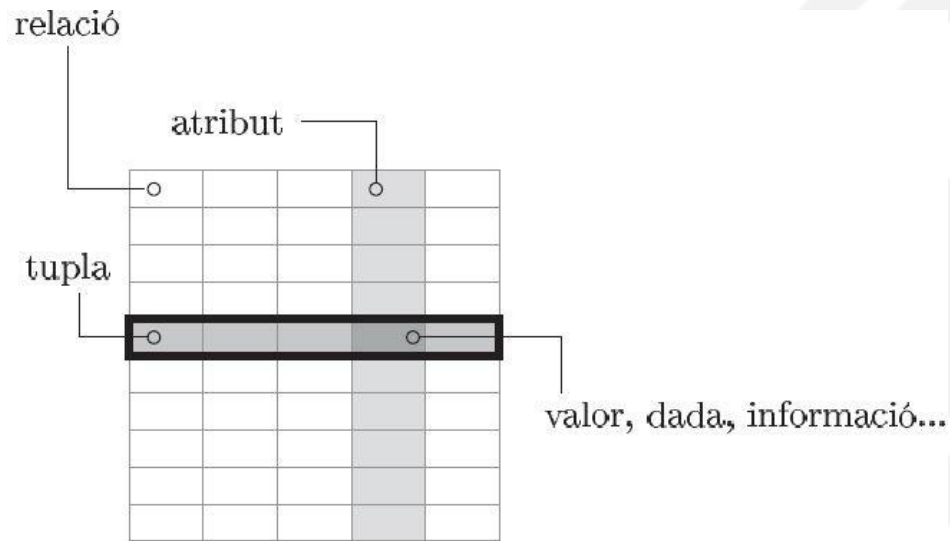
2 Llista o enumeració de noms, adreces o altres indicacions especialment ordenats.

3 1 Lligam, referència, connexió, que hom percep o imagina entre dues o més coses. *Una relació de semblança. La filosofia i la teologia tenen una relació molt estreta. Entre aquests fets no hi ha cap relació.*

2. El model relacional

En definitiva, doncs, estem parlant d'un conjunt de tuples, que són elements amb un nombre fix d'atributs.

El valor d'una tupla en un atribut és una dada, o una informació elemental.



2. El model relacional

Regla 1 (de les bases de dades relacionals)

Les tuples d'una relació tenen un nombre fix d'atributs, cada un dels quals té una mida màxima.

Amb fitxers calculats la gestió de les bases de dades es faria molt complicada, així doncs, el nombre de dades que té una relació és el nombre d'atributs multiplicat pel nombre de tuples.

La identificació de les cel·les passa per la identificació de les coordenades fila, columna.

2.2 Claus en el model relacional

Atributs requerits. Establir una restricció d'existència per a un atribut d'una relació vol dir no permetre afegir tuples a la relació si no tenen un valor en aquell atribut.

Atributs únics. Establir una restricció d'unicitat per a un atribut d'una relació vol dir no permetre afegir tuples a la relació si ja existeix en l'atribut alguna tupla amb el mateix valor que el que es pretén inserir.

No hi ha redundància entre els dos tipus de restriccions. Un atribut únic no té per què ser requerit.

Una **clau primària**, o principal, és un subconjunt dels atributs d'una relació sobre el qual s'estableixen les restriccions d'existència i d'unicitat amb la intenció d'identificar les tuples de la relació

2.2 Claus en el model relacional

En el model relacional, en els esquemes de les relacions definits en l'àlgebra relacional posarem les claus primàries al principi, subratllades

```
persona(passaport, ciutat, cognom, nom)
```

```
persona(nom, cognom, ciutat)
```

Els valors de l'atribut clau en la instància de l'entitat persona són diferents

passaport	nom	cognom	mail	ciutat
27673812M	Carme	Peralta	carmep@ionos.cat	Cadaqués
X3478937A	Carles	Sanàbria	carsan1994@gmail.com	Badalona
47548338K	Anna	Sanàbria	annasanabria@gmail.com	Badalona
45493393Z	Jesús	Hortesa	rexstat143@rediris.es	Castelldefels
C00001549	Michael	Bros	mbros1989@aol.com	San Francisco
294394950	Klauss	Stallman	kstallman@dvw.tum.de	Berlín

2.2 Claus en el model relacional

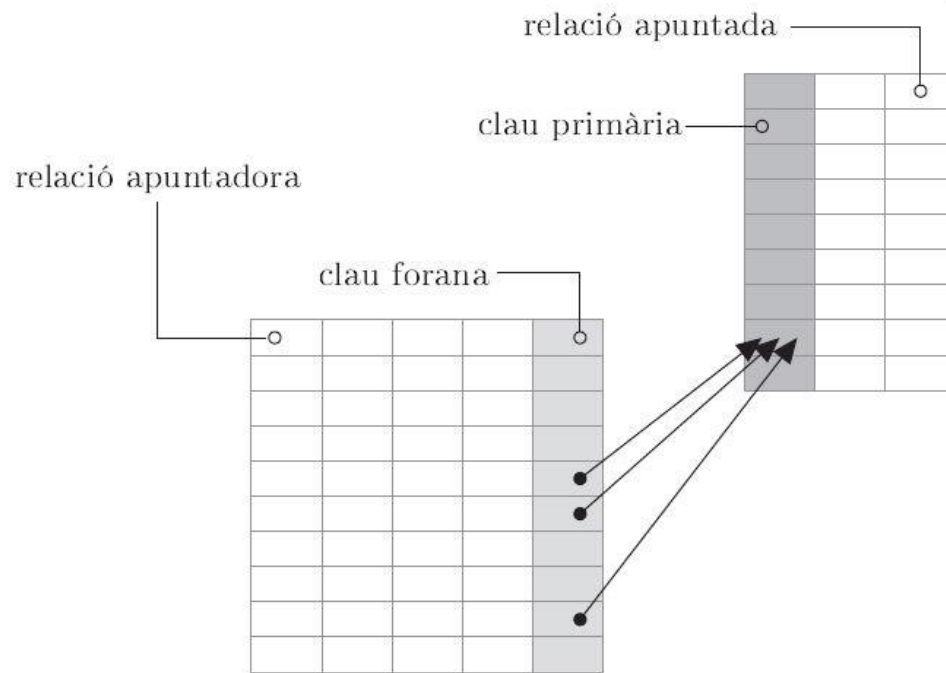
Una **clau forana**, o externa, és un subconjunt d'atributs d'una relació el domini dels quals és el conjunt de valors que conté una clau primària d'alguna altra relació de la mateixa base de dades. Són les fletxes del model ER implementades en una màquina i la raó per la qual es defineixen claus primàries.

Regla 2 (de les bases de dades relacionals)

Les claus foranes implementen les relacions 1:N d'un model ER.

Diem que una clau forana i la clau primària a la que apunta són columnes vinculades. Convé que es diguin igual. La Regla 2 és conseqüència de la Regla 1.

2.2 Claus en el model relacional

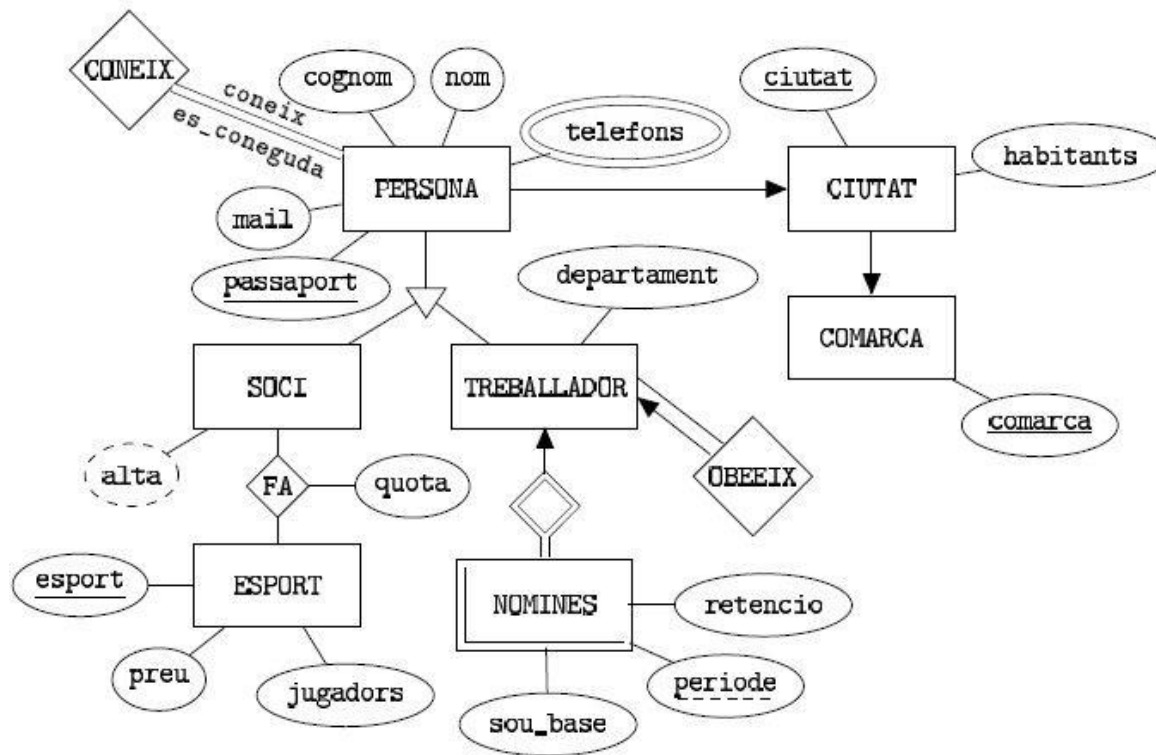


2.2 Claus en el model relacional

El model relacional d'una base de dades és la llista dels esquemes de les relacions que la componen. Només poden apuntar a relacions que ja existeixen.

El disseny del model següent és per a la base de dades del club esportiu. Gestiona les dades d'una sèrie de persones que són socis del club o bé treballadors. És necessari el número de passaport de les persones per poder-les donar d'alta a la base de dades. A més del nom, el cognom i l'adreça de correu electrònic de les persones, també es guarden diversos telèfons per a cada una d'elles, en quina ciutat viuen, quants habitants té la ciutat, i la comarca on es troba. D'altra banda, també és capaç d'establir amistats entre les persones que guarda. Les persones poden ser socis, o bé treballadors, o les dues coses alhora. Per als socis es guarda automàticament la data d'alta. I per als treballadors es guarda el departament, que pot ser administració, comercial o entrenador.

2.2 Claus en el model relacional



2.2 Claus en el model relacional

El model relacional es comença a construir a partir de les relacions que no apunten a cap altra, després les que només apuntin a les primeres, etc.

```
comarca(comarca)  
ciutat(ciutat,habitants,comarca)  
persona(passaport,nom,cognom,mail,ciutat)  
...
```

Les claus principals van al principi i subratllades, les claus foranes al final i homònimes sempre que es pugui.

2.2 Claus en el model relacional

```
comarca(comarca)  
ciutat(ciutat,habitants,comarca)  
persona(passaport,nom,cognom,mail,ciutat)  
telefon(passaport,telefon)  
coneix(coneix,es_coneguda)  
soci(passaport,alta)  
treballador(passaport,departament,obeeix)  
...
```

Les relacions que no representen una entitat no tenen clau primària, es posa la forana al davant. Les autorelacions utilitzen dues claus foranes.

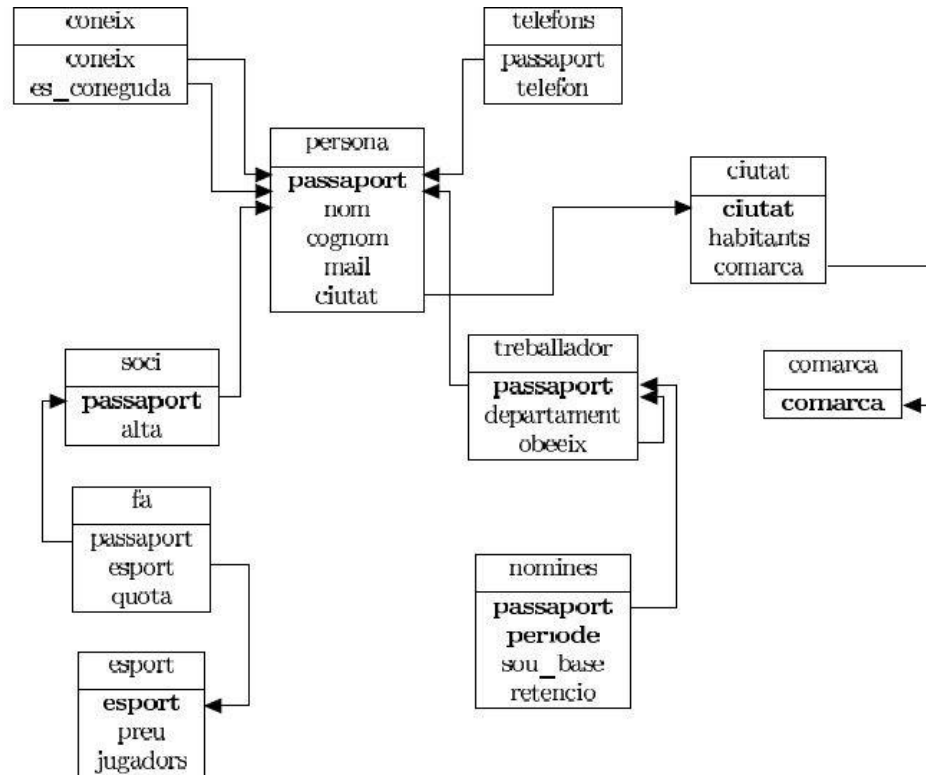
Les entitats especialitzades incorporen claus que són primàries i foranes alhora

2.2 Claus en el model relacional

Per afegir la relació fa, primer cal afegir la relació esport. S'afegeix la relació corresponent a l'entitat feble.

```
comarca(comarca)  
ciutat(ciutat,habitants,comarca)  
persona(passaport,nom,cognom,mail,ciutat)  
telefon(passaport,telefon)  
coneix(coneix,es_coneguda)  
soci(passaport,alta)  
treballador(passaport,departament,obeeix)  
esport(esport,preu,jugadors)  
fa(soci,esport,quota)  
nomines(passaport,periode,sou_base,retencio)
```

2.2 Claus en el model relacional



2.2 Claus en el model relacional

Per tant, per transformar el model ER en model relacional, cal:

model ER	claus del model relacional
entitat	clau primària en un sol atribut
relació 1:N	clau forana
relació M:N	parella de claus foranes
atribut multivalorat	clau forana i cap clau primària
especialització	la clau primària és clau forana
entitat feble	clau primària formada per dos atributs, el primer dels quals és clau forana

2.3 Operacions bàsiques

exemple 5.6. *A quin país viu la gina?*

solució $\Pi_{\text{pais}} (\sigma_{\text{nom}='gina' \wedge p.\text{ciutat}=c.\text{ciutat}} (\text{persona} \times \text{ciutat}))$

Són tres passos:

- Calcular totes les parelles possibles
- Seleccionar aquelles en què les columnes vinculades coincideixin en valor i al mateix temps satisfacin el criteri especificat
- Projectar els atributs que es demanin

2.3 Operacions bàsiques

La **unió de relacions** és com la unió de conjunts. Es representa:

$$r \cup s$$

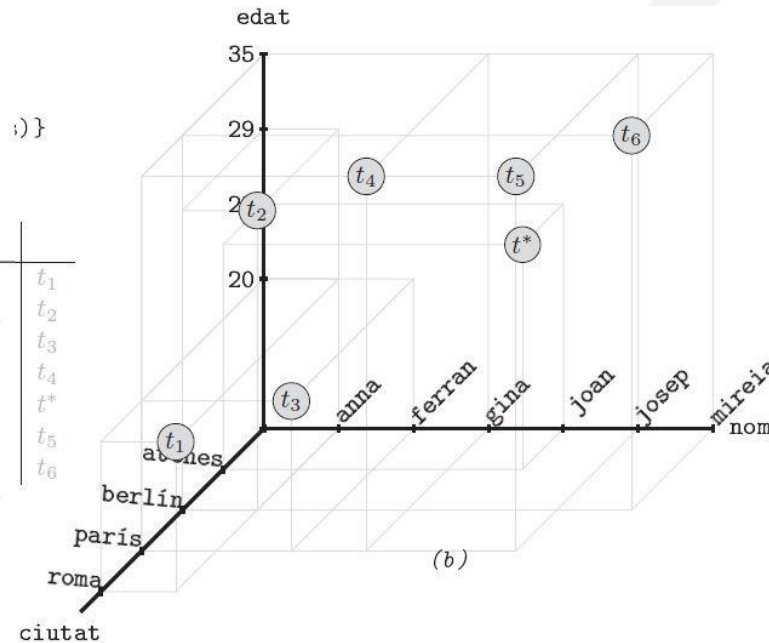
- La unió resultant de dues relacions és una relació amb les tuples de les dues. Només està definida entre relacions compatibles.
- L'esquema de la unió és la combinació de dominis menys restrictiva per als atributs. Serveix per a la inserció de noves tuples en relacions existents.

2.3 Operacions bàsiques

exemple 5.7. *Afegir en joan, de 25 anys, que viu a atenes, a la relació persona.*

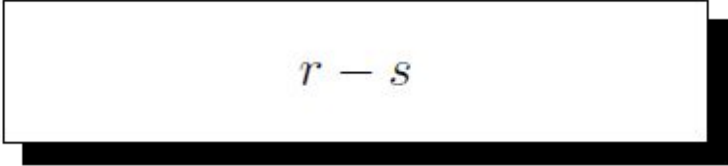
solució $\text{persona} \cup \{('joan', 25, 'atenes')\}$

nom	edat	ciutat	
anna	20	roma	t_1
anna	29	berlín	t_2
ferran	20	parís	t_3
gina	35	parís	t_4
joan	25	atenes	t_5
josep	35	parís	t_6
mireia	35	berlín	t^*



2.3 Operacions bàsiques

La **diferència de relacions** $r - s$, és la relació de les tuples de r que no són a s . Es representa:

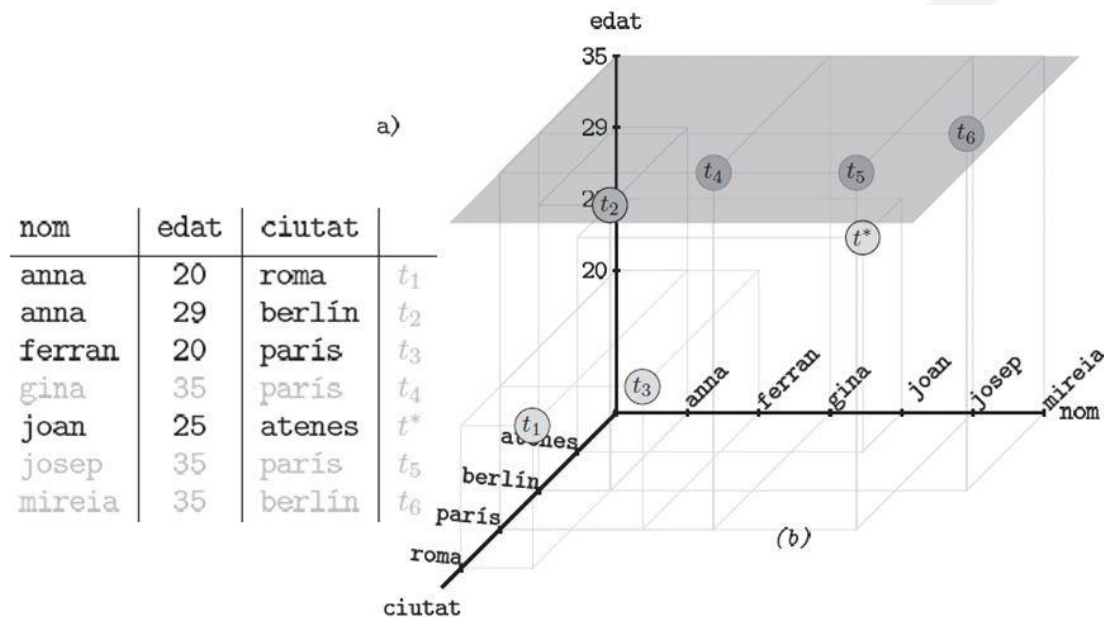
A diagram of a database relation box. It consists of a white rectangle with a black border. The expression $r - s$ is centered inside the rectangle. The box is slightly offset to the right, creating a shadow effect.
$$r - s$$

L'esquema de la diferència és l'esquema de la primera de les dues relacions d'entrada, tot i que normalment coincideixen.

2.3 Operacions bàsiques

exemple 5.8. *Eliminar totes les persones de 35 anys de la relació persona.*

solució $\text{persona} - \sigma_{\text{edat}=35}(\text{persona})$



2.3 Operacions bàsiques

El **renomament de relacions** s'expressa amb la lletra grega ρ :

$$\rho_x(A_1, A_2, \dots, A_n)(E)$$

On x és el nou nom per a l'expressió E , i (A_1, A_2, \dots, A_n) és un argument optatiu per al nom pels atributs que assignem a la relació resultant d'avaluar l'expressió relacional E , és a dir, l'esquema x .

2.3 Operacions bàsiques

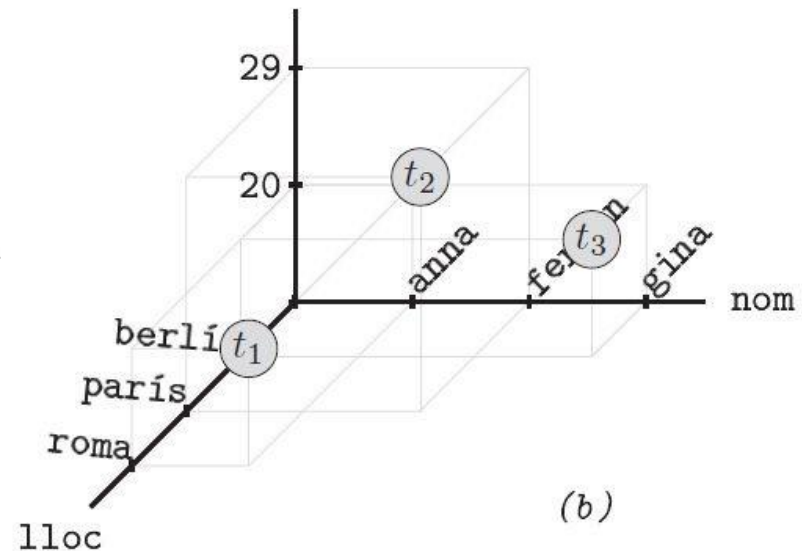
exemple 5.9. *Renomenar els atributs de la relació persona, o sigui, nom, edat i ciutat, a nom, anys i lloc.*

solució $\rho_{\text{persona}}(\text{nom}, \text{anys}, \text{lloc})(\text{persona})$

$\rho_{\text{persona}}(\text{nom}, \text{anys}, \text{lloc})(\text{persona})$

nom	anys	lloc	
anna	20	roma	t_1
ferran	29	parís	t_2
gina	20	berlín	t_3

(a)



2.3 Operacions bàsiques

nom	anys	lloc
anna	20	roma
ferran	29	parís
gina	20	berlín

exemple 5.10. *Obtenir el nom de la persona més gran.*

solució $\Pi_{\text{nom}}(\text{persona}) - \Pi_{\text{x.nom}}(\sigma_{\text{x.anys} < \text{y.anys}}(\rho_{\text{x}}(\text{persona}) \times \rho_{\text{y}}(\text{persona})))$

$\rho_{\text{x}}(\text{persona}) \times \rho_{\text{y}}(\text{persona})$

x.nom	x.anys	x.lloc	y.nom	y.any	y.lloc
anna	20	roma	anna	20	roma
anna	20	roma	ferran	29	parís
anna	20	roma	gina	20	berlín
ferran	29	parís	anna	20	roma
ferran	29	parís	ferran	29	parís
ferran	29	parís	gina	20	berlín
gina	20	berlín	anna	20	roma
gina	20	berlín	ferran	29	parís
gina	20	berlín	gina	20	berlín

2.3 Operacions bàsiques

$$\sigma_{x.\text{anys} < y.\text{anys}}(\rho_x(\text{persona}) \times \rho_y(\text{persona}))$$

x.nom	x.anys	x.lloc	y.nom	y.any	y.lloc
anna	20	roma	anna	20	roma
anna	20	roma	ferran	29	parís
anna	20	roma	gina	20	berlín
ferran	29	parís	anna	20	roma
ferran	29	parís	ferran	29	parís
ferran	29	parís	gina	20	berlín
gina	20	berlín	anna	20	roma
gina	20	berlín	ferran	29	parís
gina	20	berlín	gina	20	berlín

2.3 Operacions bàsiques

$$\Pi_{x.nom}(\sigma_{x.anys < y.anys}(\rho_x(persona) \times \rho_y(persona)))$$

$$\frac{nom}{\begin{array}{c} anna \\ gina \end{array}}$$

$$\Pi_{nom}(persona) - \Pi_{x.nom}(\sigma_{x.anys < y.anys}(\rho_x(persona) \times \rho_y(persona)))$$

$$\frac{nom}{\begin{array}{c} anna \\ ferran \\ gina \end{array}} - \frac{nom}{\begin{array}{c} anna \\ gina \end{array}} = \frac{nom}{ferran}$$

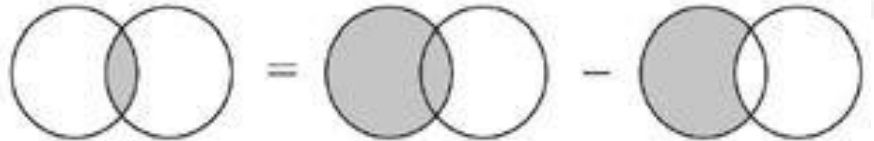
2.3 Operacions bàsiques

Les **operacions addicionals** serveixen exclusivament per fer les consultes més llegibles. Són composicions d'operacions bàsiques (intersecció, reunió natural, interna i externa, divisió, assignació...)

- **Intersecció de relacions.** És la relació formada per les tuples que estiguin en les dues relacions, r i s , que han de ser compatibles (com en la unió i la diferència).

No es tracta d'una operació bàsica:

$$r \cap s = r - (r - s)$$



2.3 Operacions bàsiques

- **Reunió natural.** És la selecció de les tuples del producte cartesià que tinguin valors coincidents en les columnes homònimes projectant-ne tan sols una.

persona

nom	edat	ciutat
gina	35	parís
josep	35	parís
mireia	35	berlín
pere	23	roma

ciutat

ciutat	país
berlín	alemanya
parís	frança
lisboa	portugal

$r \bowtie s$

persona \bowtie ciutat

nom	edat	ciutat	país
gina	35	parís	frança
josep	35	parís	frança
mireia	35	berlín	alemanya

2.3 Operacions bàsiques

- a) **Reunió interna.** És la selecció de les tuples del producte cartesià que satisfan el un predicat: $r \bowtie_{\theta} s = \sigma_{\theta}(r \times s)$

persona

nom	edat	ciutat
gina	35	parís
josep	35	parís
mireia	35	berlín
pere	23	roma

ciutat

ciutat	país
berlín	alemanya
parís	frança
lisboa	portugal

$r \bowtie_{\theta} s$

persona $\bowtie_{p.ciutat=c.ciutat}$ ciutat

nom	edat	ciutat	ciutat	país
gina	35	parís	parís	frança
josep	35	parís	parís	frança
mireia	35	berlín	berlín	alemanya

2.3 Operacions bàsiques

- a) **Reunió externa.** N'hi ha de tres tipus, per l'esquerra, consta de totes les dades de la primera de les relacions d'entrada, més les de la segona que estiguin vinculades amb alguna de la primera.

Introdueix valors null.

`persona ⋈ ciutat`

nom	edat	ciutat	ciutat	pais
gina	35	parís	parís	frança
josep	35	parís	parís	frança
mireia	35	berlín	berlín	alemanya
pere	23	roma	roma	null

$r \bowtie s$	$r \bowtie_{\subseteq} s$	$r \bowtie_{\supseteq} s$
(a)	(b)	(c)

2.3 Operacions bàsiques

persona ⋈ ciutat

nom	edat	ciutat	ciutat	pais
gina	35	parís	parís	frança
josep	35	parís	parís	frança
mireia	35	berlín	berlín	alemanya
pere	23	roma	roma	null
null	null	lisboa	lisboa	portugal

persona ⋈ ciutat

nom	edat	ciutat	ciutat	pais
gina	35	parís	parís	frança
josep	35	parís	parís	frança
mireia	35	berlín	berlín	alemanya
null	null	lisboa	lisboa	portugal

2.3 Operacions bàsiques

Les **funcions d'agregació** són funcions que retornen un valor a partir d'una col·lecció de valors. N'hi ha 5: $\text{count}(A)$, $\text{min}(A)$, $\text{max}(A)$, $\text{sum}(A)$ i $\text{avg}(A)$.

exemple 5.20. *Quants socis hi ha al club?* solució $\mathcal{G}_{\text{count}(\ast)}(\text{soci})$

exemple 5.21. *Quants socis hi ha al club que es diguin joan?*

solució $\mathcal{G}_{\text{count}(\ast)}(\sigma_{\text{nom}=\text{'joan'}}(\text{persona}) \bowtie \text{soci})$

exemple 5.22. *Com es diuen el primer i l'últim soci, alfabèticament, del club?*

solució $\mathcal{G}_{\text{min}(\text{nom}), \text{max}(\text{nom})}(\text{persona} \bowtie \text{soci})$

exemple 5.23. *Quants diners entren cada mes al club?* solució $\mathcal{G}_{\text{sum}(\text{quota})}(\text{fa})$

exemple 5.24. *Quant ingressa el club per cada esport?* solució $\text{esport } \mathcal{G}_{\text{sum}(\text{quota})}(\text{fa})$

exemple 5.25. *Quant paga cada soci?* solució $\text{passaport } \mathcal{G}_{\text{sum}(\text{quota})}(\text{fa} \bowtie \text{soci})$

2.4 Teoria de la normalització

La **teoria de la normalització** és un mètode que permet assegurar si un disseny relacional (tant directe com provinent de la traducció d'un ER) és més o menys correcte.

Es defineixen les **formes normals** com a indicadors per avaluar el grau de normalitat de les relacions, amb l'objectiu que les relacions d'un model relacional assoleixin el grau màxim de normalitat.

El procés de normalització per aconseguir augmentar la forma normal d'una relació consisteix en la descomposició de les relacions original en dues o més relacions que verifiquin un nivell de forma normal superior.

2.4 Teoria de la normalització

La **relació universal** és la que agrupa els atributs corresponents a tots els conceptes que constitueixen una base de dades. Per exemple:

Num	DataComanda	Article	Descripcio	Qtat	Preu	DataPrevista	N
22.523	25-05-2000	PC3-500	PC Pentium III a 500	5	150	1-06-2000	A
22.523	25-05-2000	PRO-15	Protector pantalla 15"	5	8	1-06-2000	A
22.524	27-05-2000	PC3-500	PC Pentium III a 500	15	145	5-06-2000	M
22.524	27-05-2000	PRO-15	Protector pantalla 15"	15	50	5-06-2000	M
22.525	27-05-2000	INK430	Cartutx de tinta 430	20	25	31-5-2000	A

2.4 Teoria de la normalització

- La **primera forma normal (1FN)** implica que cap atribut pot contenir valors no atòmics (indivisibles).
- El procés que cal seguir per assolir una 1FN és afegir tantes files com sigui necessari per a cadascun dels diferents valors del camp o camps que tinguin valors no atòmics.

Taula 2.4. Relació que té atributs multivalor i, per tant, no es troba en 1FN

Num	DataComanda	Article	Descripcio	Qtat	Preu	DataPrevista	NomProv	PaisProv	Moneda
22.523	25-05-2000	PC3-500 , PRO-15	PC Pentium III a 500 , Protector Pantalla 15"	5, 5	150, 8	1-06-2000	ARKANSAS	XINA	EUR
22.524	27-05-2000	PC3-500 , PRO-15	PC Pentium III a 500 , Protector Pantalla 15"	15, 15	145, 50	5-06-2000	MELISSA	ITÀLIA	USD
22.525	27-05-2000	INK430	Cartutx de tinta 430	20	25	31-5-2000	ARKANSAS	XINA	EUR

Taula 2.5. Relació en 1FN

Num	DataComanda	Article	Descripcio	Qtat	Preu	DataPrevista	NomProv	PaisProv	Moneda
22.523	25-05-2000	PC3-500	PC Pentium III a 500	5	150	1-06-2000	ARKANSAS	XINA	EUR
22.523	25-05-2000	PRO-15	Protector pantalla 15"	5	8	1-06-2000	ARKANSAS	XINA	EUR
22.524	27-05-2000	PC3-500	PC Pentium III a 500	15	145	5-06-2000	MELISSA	ITÀLIA	USD
22.524	27-05-2000	PRO-15	Protector pantalla 15"	15	50	5-06-2000	MELISSA	ITÀLIA	USD
22.525	27-05-2000	INK430	Cartutx de tinta 430	20	25	31-5-2000	ARKANSAS	XINA	EUR

2.4 Teoria de la normalització

- La **segona forma normal (2FN)** implica que està en 1FN i que tot atribut que no pertanyi a la clau té dependència funcional total de la clau (per cada valor de la clau existeix un, i només un, valor de cada atribut).
- El procés que cal seguir per assolir una 2FN és dividir la relació (conservant la informació i les dependències) en tantes relacions com sigui necessari de forma que cada relació verifiqui que els seus atributs no-clau tenen dependència funcional total de la clau.

Taula 2.6. Relació en 1FN

Num	DataComanda	Article	Descripció	Qtat	Preu	DataPrevista	NomProv	PaisProv	Moneda
22.523	25-05-2000	PC3-500	PC Pentium III a 500	5	150	1-06-2000	ARKANSAS	XINA	EUR
22.523	25-05-2000	PRO-15	Protector pantalla 15"	5	8	1-06-2000	ARKANSAS	XINA	EUR
22.524	27-05-2000	PC3-500	PC Pentium III a 500	15	145	5-06-2000	MELISSA	ITÀLIA	USD
22.524	27-05-2000	PRO-15	Protector pantalla 15"	15	50	5-06-2000	MELISSA	ITÀLIA	USD
22.525	27-05-2000	INK430	Cartutx de tinta 430	20	25	31-5-2000	ARKANSAS	XINA	EUR

2.4 Teoria de la normalització

Taula 2.7. Relació en 2FN que emmagatzema les comandes

COMANDA					
Num	DataComanda	DataPrevista	NomProv	PaisProv	Moneda
22.523	25-05-2000	1-06-2000	ARKANSAS	XINA	EUR
22.524	27-05-2000	5-06-2000	MELISSA	ITÀLIA	USD
22.525	27-05-2000	31-5-2000	ARKANSAS	XINA	EUR

Taula 2.8. Relació en 2FN pels articles

ARTICLE	
Article	Descripció
PC3-500	PC Pentium III a 500
PRO-15	Protector pantalla 15"
INK430	Cartutx de tinta 430

Taula 2.9. Relació en 2FN pel detall de comanda

DETALL			
Num	Article	Qtat	Preu
22.523	PC3-500	5	150
22.523	PRO-15	5	8
22.524	PC3-500	15	145
22.524	PRO-15	15	50
22.525	INK430	20	25

2.4 Teoria de la normalització

La **tercera forma normal (3FN)** implica que està en 2FN i que els atributs que no pertanyen a la clau depenen transitivament de la clau a través d'un tercer atribut (que depèn de la clau i de l'atribut, però la clau no en depèn).

El procés que cal seguir per assolir una 3FN és dividir la relació (conservant la informació i les dependències) en noves relacions més simples, de manera que cada relació verifiqui que cap dels seus atributs no-clau, depèn transitivament de la clau.

Taula 2.6. Relació en 1FN									
Num	DataComanda	Article	Descripció	Qtat	Preu	DataPrevista	NomProv	PaisProv	Moneda
22.523	25-05-2000	PC3-500	PC Pentium III a 500	5	150	1-06-2000	ARKANSAS	XINA	EUR
22.523	25-05-2000	PRO-15	Protector pantalla 15"	5	8	1-06-2000	ARKANSAS	XINA	EUR
22.524	27-05-2000	PC3-500	PC Pentium III a 500	15	145	5-06-2000	MELISSA	ITÀLIA	USD
22.524	27-05-2000	PRO-15	Protector pantalla 15"	15	50	5-06-2000	MELISSA	ITÀLIA	USD
22.525	27-05-2000	INK430	Cartutx de tinta 430	20	25	31-5-2000	ARKANSAS	XINA	EUR

2.4 Teoria de la normalització

Taula 2.10. Relació en 3FN que emmagatzema les comandes

COMANDA				
Num	DataComanda	DataPrevista	CodProv	Moneda
22.523	25-05-2000	1-06-2000	ARK	EUR
22.524	27-05-2000	5-06-2000	MEL	USD
22.525	27-05-2000	31-5-2000	ARK	EUR

Taula 2.11. Relació en 3FN pels proveïdors

PROVEIDOR		
CodProv	NomProv	PaisProv
ARK	ARKANSAS	XINA
MEL	MELISSA	ITÀLIA

2.4 Teoria de la normalització

La **quarta forma normal (4FN)** implica que està en 3FN i que l'implicant de tota dependència multivalent (un valor d'un atribut que determina un conjunt de valors d'un altre) és una clau candidata (atribut o conjunt d'atributs que permeten identificar les tuples que conté la seva extensió).

El procés que cal seguir per assolir una 4FN a partir d'una relació $R(A, B, C)$ que té una dependència multivalent entre A i B , passa per descompondre la relació R en dues relacions $R_1(A, B)$ i $R_2(A, C)$.

2.4 Teoria de la normalització

Taula 2.14. Relació en 4FN

CREDIT_EN_CURS	
Dni	Credit
10.000.000	SGBD
10.000.000	ADBD
20.000.000	PEM
20.000.000	ADBD
20.000.000	SGBD
15.000.000	PEM

Taula 2.15. Relació en 4FN

ESPORT_EN_PRACTICA	
Dni	Esport
10.000.000	Bàsquet
10.000.000	Futbol
20.000.000	Natació
20.000.000	Esgrima
15.000.000	Natació
15.000.000	Bàsquet

2.4 Teoria de la normalització

- La **cinquena forma normal (5FN)** implica que està en 4FN i que tota dependència de reunió és conseqüència de les claus candidates.
- El procés que cal seguir per assolir una 4FN és descompondre, sense pèrdua d'informació, una relació 4FN en les relacions sobre les quals es defineix la dependència de reunió, les quals estan en 5FN.

PROFESSOR			
CodiProf	Centre	Especialitat	Tasca
P1	C1	Matemàtiques	Tutor
P1	C2	Matemàtiques	Tutor
P1	C2	Informàtica	Aula Informàtica
P2	C1	Català	Coordinador
P2	C2	Castellà	Tutor

2.4 Teoria de la normalització

PCE		
CodiProf	Centre	Especialitat
P1	C1	Matemàtiques
P1	C2	Matemàtiques
P1	C2	Informàtica
P2	C1	Català
P2	C2	Castellà

PCT		
CodiProf	Centre	Tasca
P1	C1	Tutor
P1	C2	Tutor
P1	C2	Aula informàtica
P2	C1	Coordinador
P2	C2	Tutor

PET		
Professor	Especialitat	Tasca
P1	Matemàtiques	Tutor
P1	Informàtica	Aula informàtica
P2	Català	Coordinador
P2	Castellà	Tutor

La relació PROFESSOR no es troba en 5FN perquè hem trobat la dependència de reunió $\text{PROFESSOR}^*(\text{PCE}, \text{PCT}, \text{PET})$ en què aquestes tres no estan constituïdes per claus candidates de PROFESSOR.

Per tant, PROFESSOR desapareixeria per donar pas a PCE, PCT i PET, les relacions en què es basa la dependència de reunió trobada.

2.4 Teoria de la normalització

- Després d'argumentar substancialment la importància que té disposar de bases de dades normalitzades, pot semblar estrany plantejar-se que, en alguns casos, una desnormalització controlada pot ser molt útil i, fins i tot, desitjable. Però, efectivament, és així.
- La **desnormalització** és la introducció de redundàncies de forma controlada en una base de dades, per tal de fer més eficients alguns processos que, altrament, farien que globalment el rendiment del sistema resultés poc òptim.

4.1 Llenguatge de definició de dades, DDL

En el moment de creació d'una taula, cal especificar un tipus de dada per a cadascuna de les columnes, perquè totes les tuples d'una relació han de tenir el mateix tipus de dades en cada atribut. Podem distingir quatre grans grups de dades:

- Tipus de dades per gestionar informació alfanumèrica
- Tipus de dades per gestionar informació numèrica
- Tipus de dades per gestionar informació temporal
- Altres tipus de dades

Dades alfanumèriques

Les dades alfanumèriques s'emmagatzemen en tipus de dades string, que tenen menys propietats i, per tant, són menys restrictives que altres tipus de dades.

MySQL proporciona els tipus de dades següents:

CHAR (llargada): Cadena de caràcters de longitud fixa. Tots els valors de la columna han de tenir la longitud especificada (si és més curta, s'emplena amb espais en blanc i si és més llarga, es trunca).

La llargada mínima i per defecte per a una columna de tipus CHAR és d'1 caràcter, i la llargada màxima permesa és de 255 caràcters. Per indicar la llargada, cal especificar-la amb un nombre entre parèntesis, que indica el nombre de caràcters, que tindrà l'string. Per exemple CHAR(10).

Dades alfanumèriques

VARCHAR (llargada): Cadena de caràcters de longitud variable que pot ser, com a màxim, la indicada per llargada. S'emmagatzema el valor exacte (sense espais addicionals), si és més llarga, retorna un error.

La llargada màxima permesa és de 65.535 caràcters. Per indicar la llargada, cal especificar-la amb un nombre entre parèntesis, que indica el nombre de caràcters màxim, que tindrà l'string. Per exemple VARCHAR(10).

És el tipus de dades alfanumèriques més habitual en bases de dades MySQL.

Dades alfanumèriques

- **BINARY (llargada)**: És similar al tipus CHAR però emmagatzema caràcters en binari. En aquest cas, la llargada sempre s'indica en bytes.. La llargada mínima per a una columna BINARY és d'byte. La llargada màxima permesa és de 255. També hi ha el tipus **VARBINARY (llargada)** que omple amb @IOCCONTENT@ (és a dir, 0x00 en hexadecimal) els bytes que no s'omplen explícitament.
- **BLOB**. Permet contenir una quantitat gran i variable de dades de tipus binari. S'emmagatzemen en un objecte separat de la resta de columnes de la taula. Trobem:
 - TINYBLOB ($2^8 + 2$ bytes)
 - MEDIUMBLOB ($2^{24} + 3$ bytes)
 - BLOB ($2^{16} + 2$ bytes)
 - LONGBLOB ($2^{36} + 4$ bytes)

Dades alfanumèriques

TEXT: Permet contenir una quantitat gran i variable de caràcters. També s'emmagatzemen en un objecte separat de la resta de columnes de la taula. Trobem:

- TINYTEXT ($2^8 + 2$ bytes)
- TEXT ($2^{16} + 2$ bytes)
- MEDIUMTEXT ($2^{24} + 3$ bytes)
- LONGTEXT ($2^{36} + 4$ bytes)

ENUM ('cadena1', 'cadenaN'): llista prefixada de cadenes de caràcters definits en el moment de definir la columna i que es correspondran amb els valors possibles. El valor per defecte és nul, a no ser que s'especifiqui el contrari, llavors serà el primer de la llista. El nombre màxim de cadenes diferents és 65.535. Per exemple:
ENUM ('small', 'medium', 'large').

Dades alfanumèriques

SET ('cadena1', 'cadenaN'): Pot contenir zero o més valors, però tots han de pertànyer a la llista especificada en el moment de la creació. El nombre màxim de valors diferents és 64. Per exemple, es pot definir una columna de tipus SET('one', 'two'). I un element concret pot tenir qualsevol dels valors següents:

- ''
- 'one'
- 'two'
- 'one,two' (el valor 'two,one' no es preveu perquè l'ordre dels elements no afecta les llistes)

Dades numèriques

- MySQL suporta tots els tipus de dades numèriques de SQL estàndard (INTEGER; SMALL INT, DECIMAL, NUMERIC) però també (FLOAT, REAL, DOUBLE, BIT, BOOLEAN).
- INTEGER o INT(N):** emmagatzema valors enters de N dígitos visibles. Per exemple: d'un INT(4) es mostraran tan sols 4 dígitos (encara que el valor emmagatzemat sigui més gran).

Taula 1.3. Tipus de dades INTEGER

Tipus d'enter	Emmagatzemament (en bytes)	Valor mínim (amb signe / sense signe)	Valor màxim (amb signe / sense signe)
TINYINT	1	-128 / 0	127 / 255
SMALLINT	2	-32768 / 0	32767 / 65535
MEDIUMINT	3	-8388608 / 0	8388607 / 16777215
INT	4	-2147483648 / 0	2147483647 / 4294967295
BIGINT	8	-9223372036854775808 / 0	9223372036854775807 / 18446744073709551615

Dades numèriques

FLOAT, REAL i DOUBLE (E, D): emmagatzema valors reals (admeten decimals). Els dos primers s'emmagatzemen en 4 bytes i el tercer en 8. Admeten que s'especifiqui la part entera (E) i la part decimal (D), que poden ser com a màxim 30 i mai més grans que E-2. Per exemple: Si definim `FLOAT(7, 4)` i hi volem emmagatzemar 999.00009, es convertirà en 999.0001.

DECIMAL i NUMERIC (T, D): emmagatzema valors reals de punt fix, és a dir, sense arrodonir. Si el valor introduït no és adequat emet un error. Permeten especificar el total de dígit (T), per defecte 10 i com a màxim 65 i la quantitat de dígit decimals (D). Per exemple: si definim `NUMERIC(5, 2)` podria contenir valors des de -999.99 fins a 999.99

Dades numèriques

BIT (M): emmagatzema M bits, des d'1 bit (per defecte) fins a 64. És sinònim de TINYINT(1). Per exemple, si afegim b'1010' a un camp definit BIT(6) el valor emmagatzemat serà b'001010', afegeix zeros a l'esquerra per completar-lo.

BOOLEAN o BOOL: emmagatzema dades booleans (cert o fals). És sinònim de BIT(1). El zero es considera fals, i l'u es considera cert.

Modificadors de tipus numèric: paraules clau que afegides a la definició d'una columna numèrica (entera o real) condiciona els valors que contindran:

- UNSIGNED: només admet valors de tipus numèric no negatiu
- ZEROFILL: afegirà zeros a l'esquerra per completar el total de dígit, si cal
- AUTO_INCREMENT: afegeix el valor més alt de la columna incrementat en 1 en comptes del valor 0 o nul.

Dades temporals

DATE: emmagatzema dates en format AAAA-MM-DD.

DATETIME: emmagatzema combinacions de dies i hores en format AAAA-MM-DD HH:MM:SS.

TIMESTAMP: emmagatzema l'hora i la data actuals en un moment determinat en el mateix format que DATETIME.

TIME: emmagatzema l'hora en format HH:MM:SS. També pot emmagatzemar la diferència de temps entre dos moments.

YEAR: emmagatzema anys en format AAAA de tipus BYTE o 'AAAA' de tipus string.

Altres tipus de dades

- En MySQL hi ha extensions que permeten emmagatzemar altres tipus de dades, com les dades poligonals.
- Així doncs, MySQL permet emmagatzemar dades de tipus objecte poligonal i dóna implementació al model geomètric de l'estàndard Open-GIS.
- Per tant, podem definir columnes MySQL de tipus Polygon, Point, Curve o Line, entre d'altres.

Manipulació i control de dades

- CREATE, ALTER, DROP i TRUNCATE
- Selecció:
 - SELECT, DISTINCT
- Modificació:
 - INSERT
 - UPDATE
 - DELETE

Selección de filas

- La cláusula WHERE permite filtrar las filas de la relación resultante.
 - La condición de filtrado se especifica como un predicado.
- El predicado de la cláusula WHERE puede ser simple o complejo
 - Se utilizan los conectores lógicos AND (conjunción), OR (disyunción) y NOT (negación)
- Las expresiones pueden contener
 - Predicados de comparación
 - BETWEEN / NOT BETWEEN
 - IN / NOT IN (con y sin subconsultas)
 - LIKE / NOT LIKE
 - IS NULL / IS NOT NULL
 - ALL, SOME/ANY (subconsultas)
 - EXISTS (subconsultas)

Funciones de columna

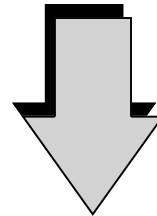
- Por defecto las funciones se aplican a todas las tuplas resultantes de la consulta.
- Podemos agrupar las tuplas resultantes para poder aplicar las funciones de columna a grupos específicos utilizando la cláusula **GROUP BY**.
- En la cláusula **SELECT** de consultas que utilizan funciones de columna solamente pueden aparecer funciones de columna.
 - En caso de utilizar **GROUP BY**, también pueden aparecer columnas utilizadas en la agrupación.
- Adicionalmente se pueden aplicar condiciones sobre los grupos utilizando la cláusula **HAVING**.

Funciones de columna

- Cálculo del total → **SUM (expresión)**
- Cálculo de la media → **AVG (expresión)**
- Obtener el valor mínimo → **MIN (expresión)**
- Obtener el valor máximo → **MAX (expresión)**
- Contar el número de filas que satisfacen la condición de búsqueda → **COUNT (*)**
 - Los valores NULL SI se cuentan.
- Contar el número de valores distintos en una columna → **COUNT (DISTINCT nombre-columna)**
 - Los valores NULL NO se cuenta.

Funciones de columna

```
SELECT      SUM(SALARY) AS SUM,  
            AVG(SALARY) AS AVG,  
            MIN(SALARY ) AS MIN,  
            MAX(SALARY) AS MAX,  
            COUNT(*)      AS COUNT,  
            COUNT(DISTINCT WORKDEPT) AS DEPT  
FROM        EMPLOYEE
```



SUM	AVG	MIN	MAX	COUNT	DEPT
873715.00	27303.59375000	15340.00	52750.00	32	8

GROUP BY

Necesito conocer los salarios de todos los empleados de los departamentos A00, B01, y C01. Además, para estos departamentos quiero conocer su masa salarial.



```
SELECT WORKDEPT, SALARY
FROM EMPLOYEE
WHERE WORKDEPT IN ('A00', 'B01', 'C01')
ORDER BY WORKDEPT
```



WORKDEPT	SALARY
A00	52750.00
A00	46500.00
A00	29250.00
B01	41250.00
C01	38250.00
C01	23800.00
C01	28420.00

```
SELECT WORKDEPT, SUM(SALARY) AS SUM
FROM EMPLOYEE
WHERE WORKDEPT IN ('A00', 'B01', 'C01')
GROUP BY WORKDEPT
ORDER BY WORKDEPT
```



WORKDEPT	SUM
A00	128500.00
B01	41250.00
C01	90470.00



enguaje SQL

GROUP BY-HAVING

Ahora sólo quiero ver los departamentos
cuya masa salarial sea superior a 50000



```
SELECT WORKDEPT, SUM(SALARY) AS SUM
FROM EMPLOYEE
WHERE WORKDEPT IN ('A00', 'B01', 'C01')
GROUP BY WORKDEPT
ORDER BY WORKDEPT
```



WORKDEPT	SUM
A00	128500.00
B01	41250.00
C01	90470.00

```
SELECT WORKDEPT, SUM(SALARY) AS SUM
FROM EMPLOYEE
WHERE WORKDEPT IN ('A00', 'B01', 'C01')
GROUP BY WORKDEPT
HAVING SUM(SALARY) > 50000
ORDER BY WORKDEPT
```



WORKDEPT	SUM
A00	128500.00
C01	90470.00

Triggers

```
CREATE TRIGGER ins_sum  
BEFORE INSERT ON account  
FOR EACH ROW SET @sum = @sum + NEW.amount;
```


Gràcies per la vostra col·laboració!

Laia.Subirats@eurecat.org



**Ajuntament
de Barcelona**



**Barcelona
Activa**