AS - Teoria 3

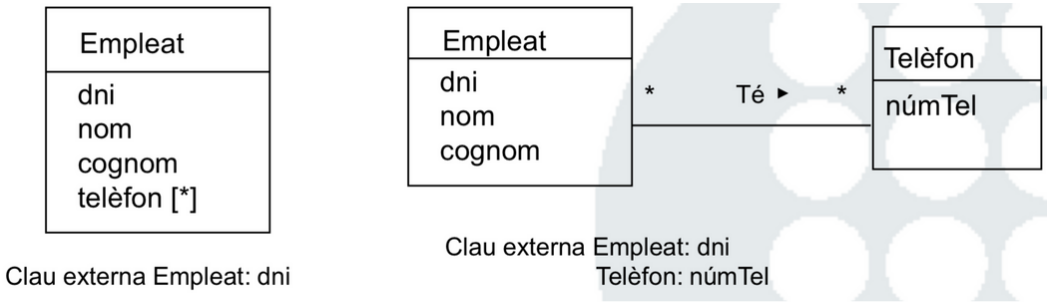
Data Layer Design

3.4.1. Relational databases design

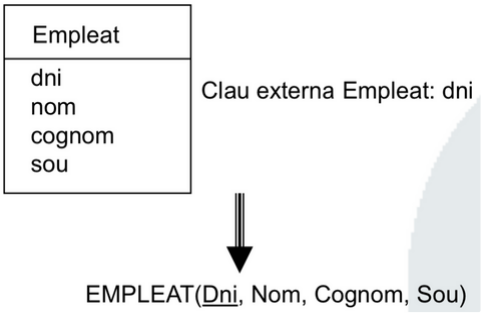
Alias → **Introducció al disseny de bases de dades relacionals.** Transformació de UML a model relacional.

Transformacions inicials del model de partida

- Per poder aplicar les transformacions que veurem més endavant cal que tots els atributs siguin **univaluats** (no poden ser multivaluats).
- Si aquest requisit no es compleix cal **transformar** el model de partida de manera que es compleixi:
  - Transformar els atributs **multivaluats en associacions**.
    - Exemple de transformació d'un atribut multivaluat:



Transformació de les classes d'objectes i els seus atributs



- Cada classe d'objectes es transforma en una relació
  - La clau **primària** de la relació serà la clau **externa** de la classe d'objectes.
  - Els atributs de la relació seran els atributs de la classe d'objectes

Observació: és possible que calgui **afegir atributs** a la relació quan es faci la transformació de les associacions on participa la classe (es veurà a la transformació de les associacions).

- Pot ser que una classe d'objectes no tingui clau externa
  - Cal afegir un identificador "*artificial*"
  - Els SGBDs tenen mecanismes per assignar valors a aquests identificadors

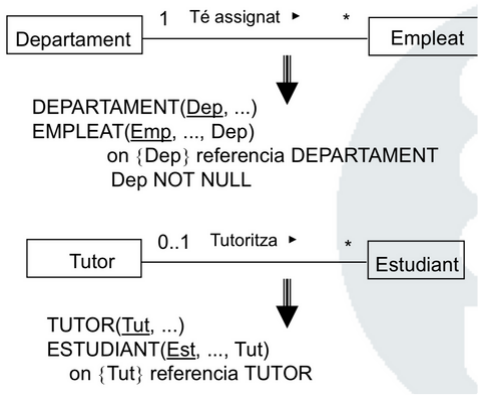
Transformació de les associacions binàries

- Per transformar una associació (binària o n-ària) cal prèviament haver transformat totes les classes d'objectes que associa.
- Casos a considerar per les associacions binàries: Cas un a molts, Cas un a un, Cas molts a molts.

> Cas un a molts

Cas un a molts: la multiplicitat és 0..1 o 1 a l'extrem 'un' i * a l'extrem 'molts' de l'associació.

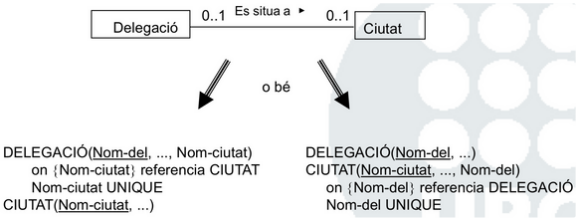
La transformació consisteix en afegir una clau forana a la relació que correspon a la classe de l'extrem 'molts' de l'associació per tal de referenciar a l'altra relació.



> Cas un a un

Cas un a un significa que la multiplicitat és 0..1 a ambdós extrems de l'associació.

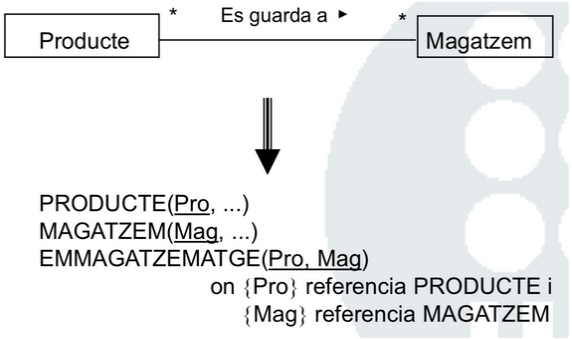
La transformació consisteix en afegir a qualsevol de les dues relacions (corresponents a les classes associades) una clau forana que referenciï a l'altra relació.



> Cas molts a molts

Cas molts a molts significa que la multiplicitat és * a ambdós extrems de l'associació.

La transformació consisteix en definir una nova relació. La seva clau primària estarà formada pels atributs de la clau primària de les relacions corresponents a les classes dels dos extrems de l'associació.

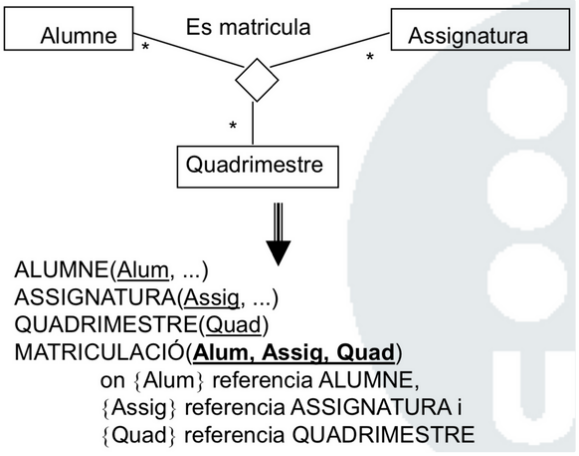


Transformació de les associacions n-àries

Per transformar una associació (binària o n-ària) cal prèviament haver transformat totes les classes d'objectes que associa.

- Les associacions n-àries sempre es transformen en una **nova relació que conté els atributs que formen la clau de les n classes associades.**
- Analitzarem alguns casos que es poden donar en associacions ternàries i després descriurem el cas general: Ternàries molts-molts-molts, Ternàries molts-molts-un, Ternàries molts-un-un, Transformació general de les n-àries.

> Ternàries molts-molts-molts



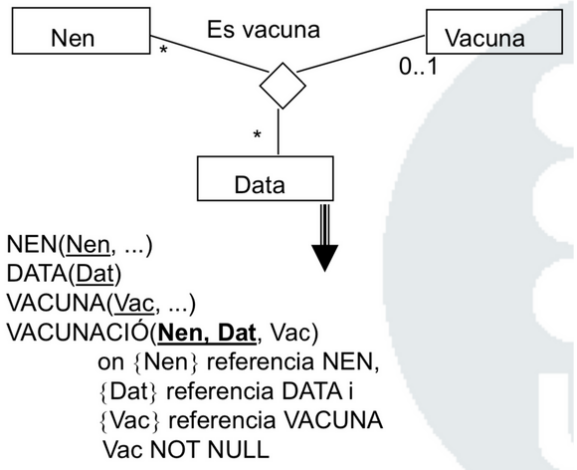
En el cas molts-molts-molts la multiplicitat és * als tres extrems de l'associació.

La clau primària de la nova relació està formada pels atributs de les claus de les classes corresponents als tres extrems de l'associació.

> Ternàries molts-molts-un

En el cas molts-molts-un la multiplicitat és * als dos extrems 'molts' i 0..1 a l'extrem 'un' de l'associació.

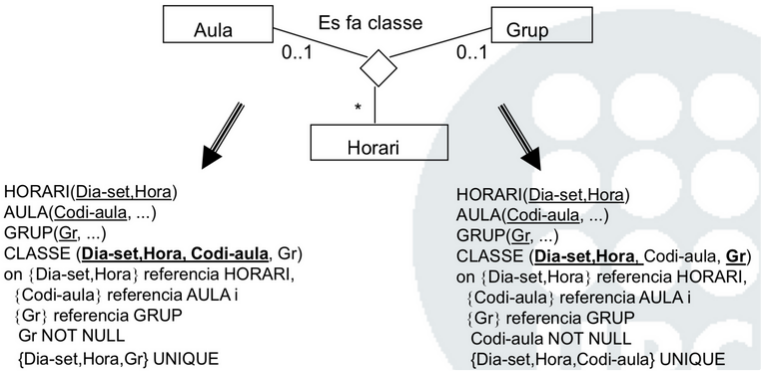
La clau primària està formada pels atributs de les claus de les classes corresponents als dos extrems 'molts' de l'associació.



> Ternàries molts-un-un

En el cas molts-un-un la multiplicitat és * al l'extrem 'molts' i 0..1 als extrems 'un' de l'associació.

La relació té dues claus candidates. Cadascuna està formada pels atributs de les claus de dues de les classes associades una de les quals és necessàriament la del costat 'molts'.



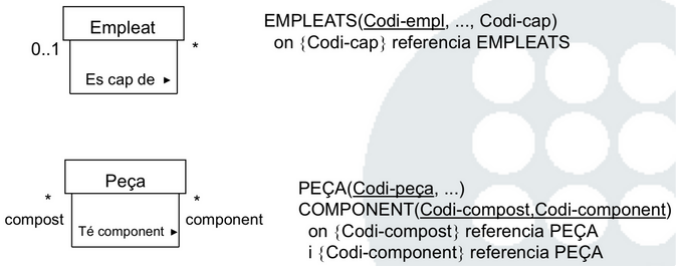
> Transformació general de les n-àries

En tots els casos, la transformació d'una interrelació n-ària consistirà en una **nova relació** que conté tots els atributs que formen les claus de les n classes associades.

- **Si totes les classes estan connectades amb 'molts'**, la clau primària de la nova relació estarà formada per tots els atributs que formen les claus de les n classes associades.
- **Si una o més classes estan connectades amb 'un'**, la clau primària de la nova relació estarà formada per les claus de n-1 de les classe associades amb la condició de que la classe, la clau de la qual no s'ha inclòs, ha de ser una de les que està connectada amb 'un'. Pot haver-hi diverses claus candidates.

Transformació de les associacions recursives

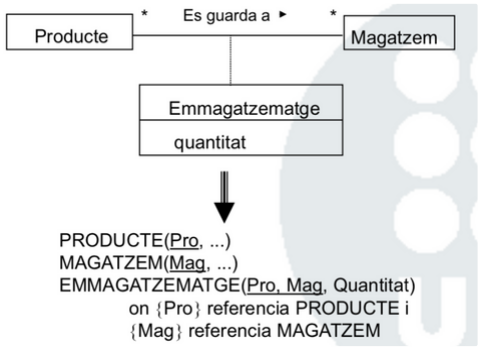
- Les associacions recursives es transformen de la mateixa manera que la resta d'associacions, és a dir, que cal tenir en compte si són binàries o n-àries i també la seva multiplicitat i aplicar les transformacions corresponents.
- Exemples:



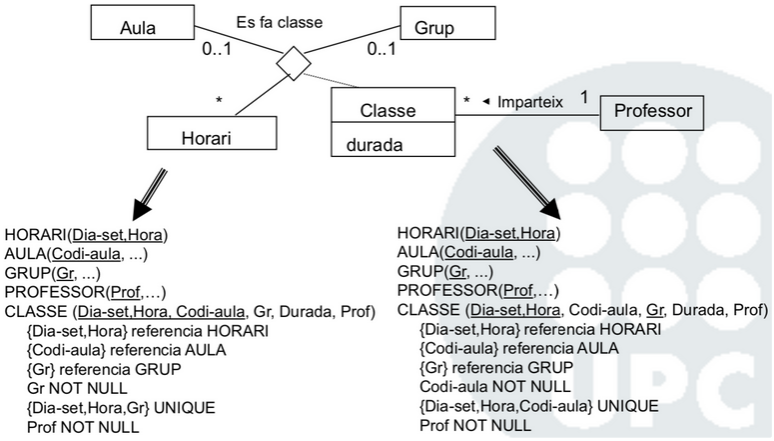
Transformació de les classes associatives

- La transformació de la seva associació és, alhora, la transformació de la classe associativa.
- Si la classe associativa té atributs, aleshores s'afegeixen com a atributs de la relació corresponent a la seva transformació.

- Exemple 1:



- Exemple 2:



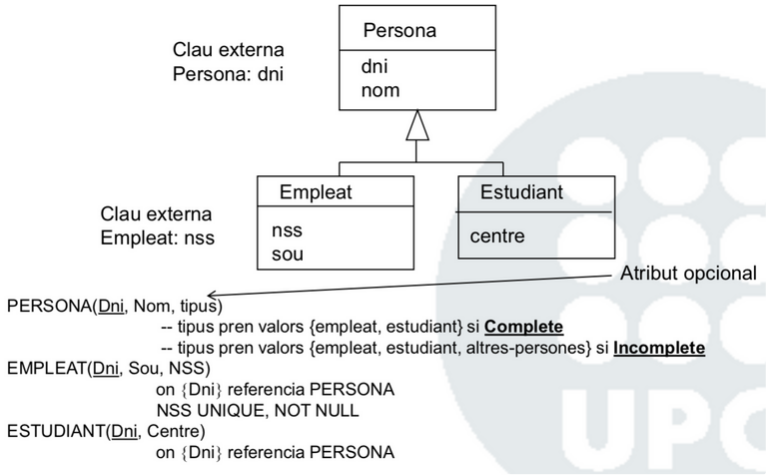
Transformació de la generalització/especialització

Hi ha tres alternatives de traducció

- Class table inheritance: Una taula per la classe i una ser cada subclasse
- Concrete table inheritance: Una taula per cada subclasse
- Single table inheritance: Una única taula

> Class table inheritance - Complete o Incomplete

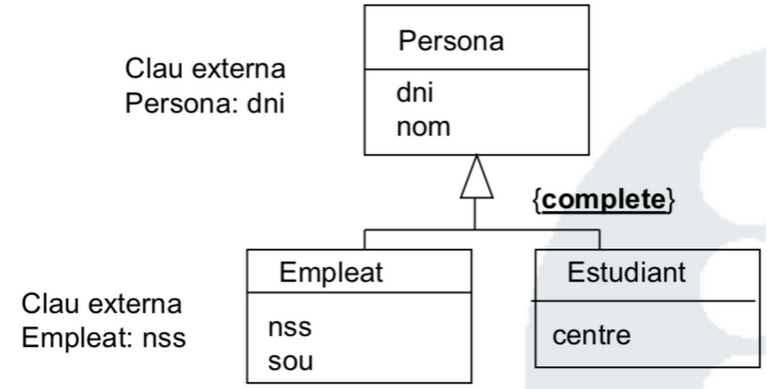
Una taula per la classe i una ser cada subclasse



PERSONA(Dni, Nom, tipus)
-- tipus pren valors {empleat, estudiant} si **Complete**
-- tipus pren valors {empleat, estudiant, altres-persones} si **Incomplete**
EMPLEAT(Dni, Sou, NSS)
on {Dni} referencia PERSONA
NSS UNIQUE, NOT NULL
ESTUDIANT(Dni, Centre)
on {Dni} referencia PERSONA

> Concrete table inheritance - Complete

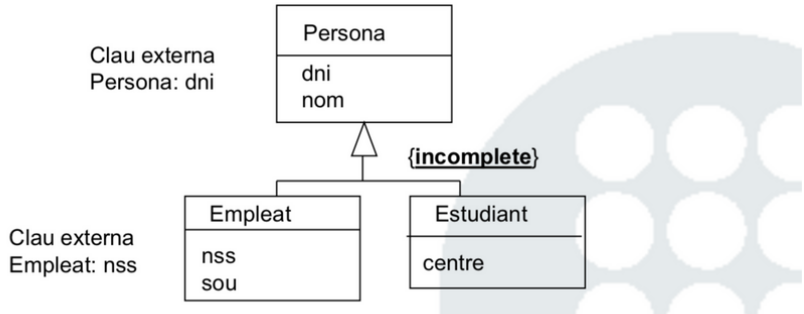
Una taula per cada subclasse



EMPLEAT(Dni, Nom, Sou, NSS)
NSS UNIQUE, NOT NULL
ESTUDIANT(Dni, Nom, Centre)

> Concrete table inheritance - Incomplete

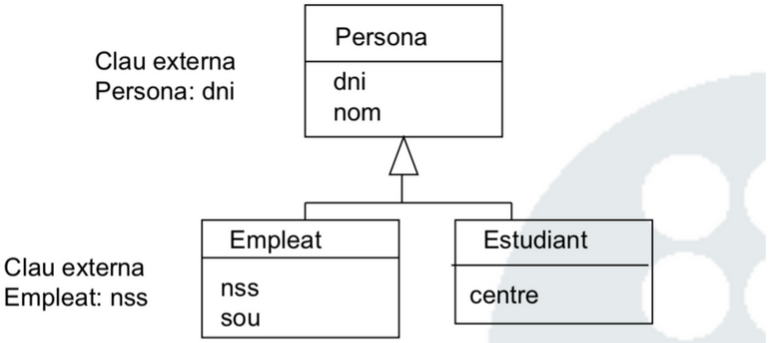
Una taula per cada subclasse i una pels objectes que no estàn a cap de les subclasses.



EMPLEAT(Dni, Nom, Sou, NSS)
NSS UNIQUE, NOT NULL
ESTUDIANT(Dni, Nom, Centre)
ALTRES-PERSONES(Dni, Nom)
-- aquesta darrera taula conté persones que no són ni empleats ni estudiants

> Single table inheritance – Complete o Incomplete

Una única taula.



PERSONA(Dni, Nom, tipus, NSS, Sou, Centre)
-- tipus pren valors {empleat, estudiant} si **Complete**
-- tipus pren valors {empleat, estudiant, altres-persones} si **Incomplete**
NSS UNIQUE – té valor únic entre els valors no nuls de l'atribut

Consideracions finals

- Les relacions corresponents a classes temporals com ara Data, Hora, Any, etc que no tenen atributs a part de la clau es poden eliminar.
- Hi ha restriccions que no es poden controlar mitjançant la definició de claus primàries i foranes de les relacions i que cal controlar amb altres mecanismes ('triggers', 'assertions', o dins dels programes).
    - Multiplicitats diferents de les considerades en aquestes transparències.
    - Restricció 'disjoint' en alguns dels dissenys presentats (ex. Concrete table inheritance). Ex: assegurar que no hi ha cap persona que estigui en dues taules.
    - Restricció 'complete' en alguns dels dissenys presentats (ex. Class table inheritance). Ex: assegurar que una persona que està a la taula persones està també en o bé la taula empleats o bé en la taula estudiants.
    - Restriccions textuals no expressables gràficament en UML (a part de la clau externa) poden requerir altres mecanismes de control.
    - Atributs/associacions derivades es poden materialitzar i mantenir actualitzats/des (amb triggers o des dels programes), calcular quan es necessiten (vistes, o des dels programes).