Disseny de la Capa de Gestió de Dades

- Introducció
- Tecnologia de les Bases de Dades Relacionals
- Estratègies de gestió de la persistència
- Generació automàtica de la persistència
  - Patró Traductor de Dades (*Data Mapper)*
  - Variant: operacions de consulta a la capa de dades
- Disseny directe de la persistència
  - Disseny directe d'operacions
    - Controlador de la Capa de Domini amb lògica simple
    - Controlador de la Capa de Domini com a intermediari
- Exemple
- Disseny de les capes de domini i gestió de dades, resum
- Bibliografia

Què és i com s'aconsegueix la persistència?

- **Persistència:**
  - és la capacitat que molts sistemes software requereixen per emmagatzemar i obtenir dades usant un sistema d'emmagatzematge permanent.
- **Els objectes es poden fer persistents en**:
  - Bases de dades orientades a objectes
  - Bases de dades relacionals
  - Bases de dades objecte-relacional
  - etc.

Com hi influeix la tecnologia de les Bases de Dades?

Dependència tecnològica:

- Propietats que es volen assolir (requisits no funcionals)
- Recursos tecnològics disponibles
  - família del llenguatge de programació
  - ***família del sistema gestor de bases de dades (SGBD)***

Determina:

- L'arquitectura del sistema software i els patrons que s'usaran

⇓

El disseny que fem depèn del SGBD utilitzat

En aquesta presentació ens centrem en l'estudi de la persistència usant SGBD relacionals

Tecnologia de les Bases de Dades Relacionals
Components de dades

- Esquema de base de dades:
  Unitat administrativa per agrupar components.
- Relacions (taules):
  Permeten emmagatzemar informació físicament.
  Un esquema de relació es denota per: $R(Atr_1, Atr_2, \ldots, Atr_n)$
- Restriccions:
  Estableixen condicions que la base de dades ha de satisfer.
  Són gestionades pel propi SGBD.
- Vistes:
  Són relacions derivades (calculades), no emmagatzemades físicament.
  El seu esquema i contingut es deriven a partir d'una consulta relacional.

Tecnologia de les Bases de Dades Relacionals
Components de procés

– Procediments emmagatzemats (*stored procedures*):
  Funcions que s'emmagatzemen a la BD i es tracten com uns objectes més de la BD.

– Disparadors (*triggers*):
  Són regles ECA (Esdeveniment, Condició, Acció) que permeten executar una Acció quan es produeix un Esdeveniment i se satisfà la Condició.

Els exemples que posarem estan fets en el SGBD Informix

Tecnologia de les Bases de Dades Relacionals
Disseny lògic de la base de dades

- **Traducció del diagrama de classes**
  - Cas Domain Model: partim del diagrama normalitzat (sense les classes provinents de l'aplicació dels patrons –Estat, etc.)
  - Cas Transaction Script: partim del diagrama d'especificació
- **Tractament de les restriccions d'integritat i informació derivada**
  - Les responsabilitats d'aquesta mena assignades a la capa de gestió de dades, s'implementen fent ús dels elements proporcionats per la tecnologia relacional (fins ara ho fèiem només amb les RI de clau, ara hi ha més elements al nostre abast)
- **Assignació de comportament**
  - Les responsabilitats (pre/post) de les operacions d'especificació que es van assignar a la capa de gestió de dades, s'implementen fent ús dels elements proporcionats per la tecnologia relacional

Eventualment, es pot decidir implementar íntegrament a la capa de dades alguna responsabilitat assignada conjuntament a les capes de domini + dades (v. Tema 4)

Tecnologia de les Bases de Dades Relacionals
Disseny lògic de la BD

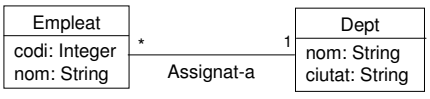
- **L'esquema conceptual** conté **classes d'objectes**
- Els **SGBDs relacionals** implementen **taules relacionals** del tipus:
  taula1 (atr1, atr2, …, atrn), on atr1 = clau primària i atrn = clau forana

Disseny lògic

- Traducció dels elements de l'esquema conceptual a components implementables pels SGBDs relacionals.
  - Classes → taules
  - Associacions entre classes d'objectes → taules, atributs, vistes
  - Aspectes no contemplats per les taules relacionals:
    - associacions *n*-àries, amb *n* > 2
    - classes associatives

    } Tractament ja conegut; transformacions ja realitzades si usem Domain Model
    - associacions binàries * - *
    - jerarquies d'especialització (i, per tant, herència)
    - identificadors interns
    - operacions associades a les classes d'objectes

Tecnologia de les Bases de Dades Relacionals
Disseny lògic de la BD, associacions binàries



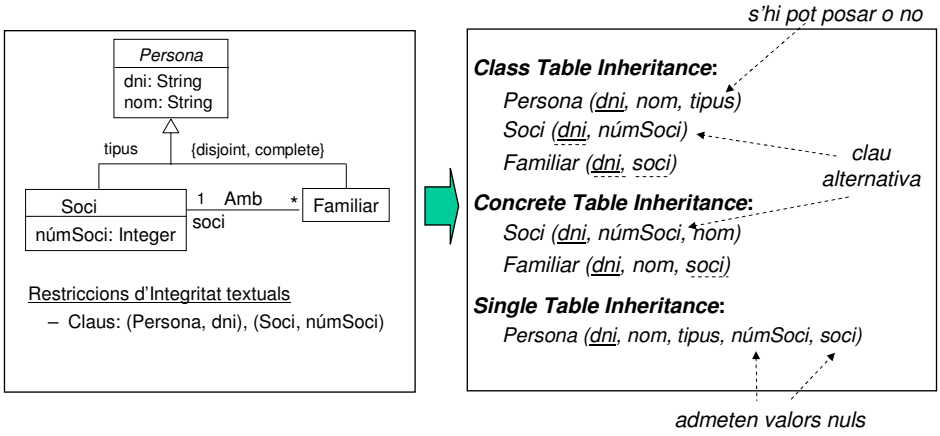
Es tradueix a:
Dept (nom-d, ciutat)
Empleat (codi, nom-emp, nom-d)

Es tradueix a:
Interval (data, inici)
Pista (número)
Manteniment (núm-pista, data, inici)

Tecnologia de les Bases de Dades Relacionals

Disseny lògic de la BD, jerarquies d'especialització (1)

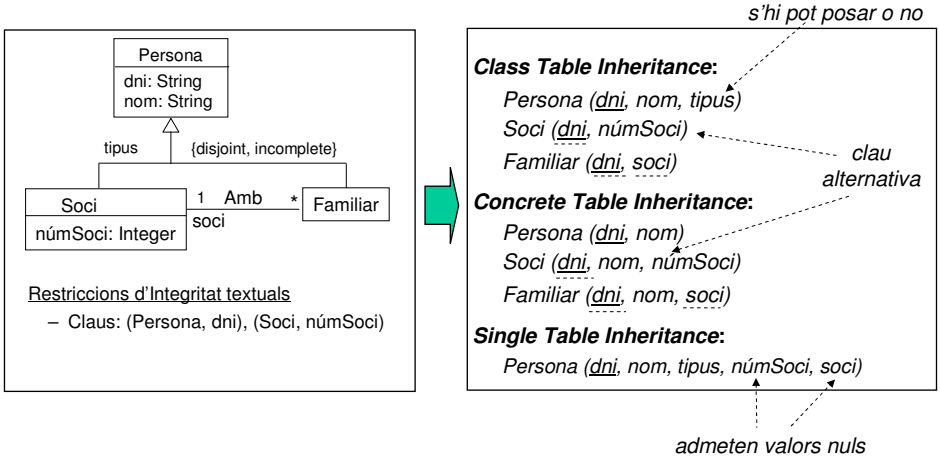
La traducció depèn de fins a quin nivell es vol col·lapsar o no la jerarquia



Tecnologia de les Bases de Dades Relacionals

Disseny lògic de la BD, jerarquies d'especialització (1, variant)

La traducció depèn de fins a quin nivell es vol col·lapsar o no la jerarquia



Tecnologia de les Bases de Dades Relacionals
Disseny lògic de la BD, jerarquies d'especialització (2)

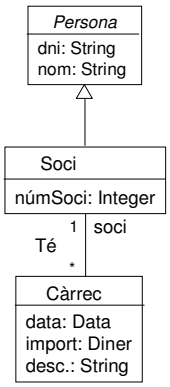
Estratègia	Avantatges	Desavantatges
Class Table Inheritance	Simple Canviable	Poc eficient (múltiples accessos per objecte)
Concrete Table Inheritance	Eficient (un accés per objecte)	Poc canviable (propagació de canvis fets a les superclasses)
Single Table Inheritance	Eficient (un accés per objecte) Canviable	Pèrdua d'espai (però la BD pot ajudar)

No hi ha una estratègia clarament millor

- consulteu l'administrador de Bases de Dades!

Tecnologia de les Bases de Dades Relacionals
Disseny lògic de la BD, classes sense clau externa

- Cal afegir alguna clau externa "artificial"
  - Normalment acostuma a ser una clau mantinguda pel propi sistema



Restriccions d'Integritat textuals

- Claus: (Persona, dni), (Soci, númSoci)

La taula Càrrec es tradueix a:

Càrrec(id-càrrec, data, import, desc., soci)

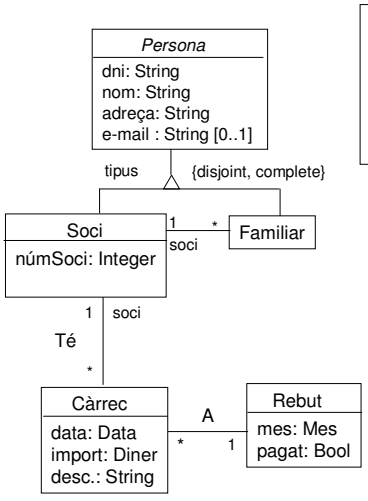
Identificador "artificial"

Tecnologia de les Bases de Dades Relacionals
Tractament de les restriccions d'integritat

- Els SGBD relacionals proporcionen diverses funcionalitats per tractar les restriccions d'integritat:
  - restriccions de columna
    - not null
    - distinct
    - unique
    - primary key
    - references taula (foreign key)
    - check (condició)
  - etc.

Podem assignar directament al SGBD relacional les responsabilitats corresponents (v. Tema 4)

Tecnologia de les Bases de Dades Relacionals
Tractament de les restriccions d'integritat, exemple



Persona (dni, nom, adreça, e-mail)
Familiar (dni, soci)
Soci (dni, númSoci)
Rebut (dniSoci, mes, pagat)
Càrrec(id-càrrec, data, import, descripció, soci, mes)

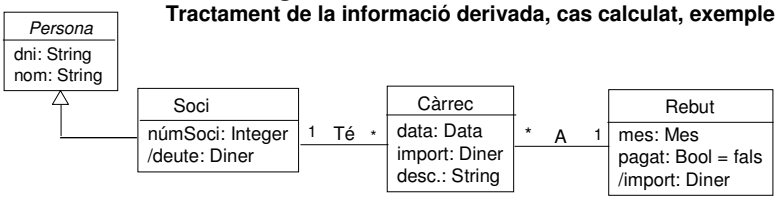
Restriccions d'Integritat textuals

- Clau: (Persona, dni)
  => Clau primària de *Persona*: *dni*
- Clau: (Soci, númSoci)
  => Clau primària de *Soci*: *dni* i clau alternativa, númSoci: unique(númSoci)
- No hi pot haver dos rebuts amb el mateix soci i mes
  => Clau primària de *Rebut*: (*dniSoci*, *mes*)
- Tots els càrrecs d'un rebut tenen una data dins de mes
  => Check de *Càrrec*: check(month(data) = mes)

Tecnologia de les Bases de Dades Relacionals
Tractament de la informació derivada

- Els atributs i les associacions derivats es poden **calcular** o **materialitzar**
- Si es *calcula*:
  - Cal assignar a algú la responsabilitat de calcular el valor d'aquesta informació.
    - A les operacions de la capa de domini
    - A la capa de gestió de dades (al propi SGBD), mitjançant **vistes** que, quan es consulten, proporcionen automàticament la informació especificada.
- Si es *materialitza*:
  - Cal assignar a algú la responsabilitat de mantenir consistent el valor d'aquesta informació.
    - A les operacions de la capa de domini
    - A la capa de gestió de dades (al propi SGBD), normalment mitjançant **disparadors** (*triggers)*
- En la decisió hi influeix el temps de càlcul, la freqüència d'accés i l'espai ocupat.

Tecnologia de les Bases de Dades Relacionals
Tractament de la informació derivada, cas calculat, exemple



Deute = suma import de rebuts no pagats
Import = suma import dels càrrecs

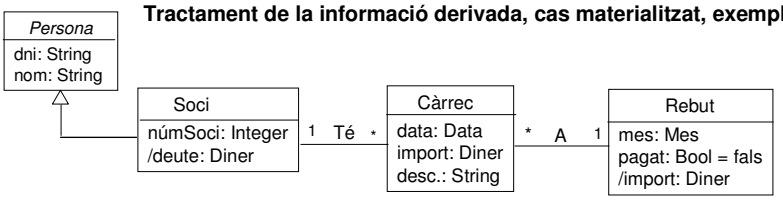
Soci (dni, númSoci)
Rebut (dniSoci, mes, pagat)
Càrrec(id-car, data, import, desc., soci, mes)

```
CREATE VIEW import-rebut [dniSoci, mes, imp] AS
SELECT   r.dniSoci, r.mes, sum(c.import)
FROM     Càrrec c, Rebut r
WHERE    c.soci = r.dniSoci AND
         c.mes = r.mes
GROUP BY dniSoci, mes
```

```
CREATE VIEW deute-soci [dniSoci, deute] AS
SELECT   s.dni, sum(c.imp)
FROM     Soci s, Càrrec c, Rebut r
WHERE    s.dniSoci = c.soci AND
         c.soci = r.dniSoci AND
         c.mes = r.mes AND
         r.pagat='F'
GROUP BY dniSoci
```

Tecnologia de les Bases de Dades Relacionals

Tractament de la informació derivada, cas materialitzat, exemple



Deute = suma import de rebuts no pagats
Import = suma import dels càrrecs

Soci (dni, númSoci, deute)
Rebut (dniSoci, mes, pagat, import)
Càrrec(id-car, data, import, desc., soci, mes)

Dues alternatives:

- Assignar la responsabilitat a la capa de domini *(tal i com ho hem fet durant el curs)*
- Assignant la responsabilitat de materialitzar al SGBD:
  - *Mitjançant disparadors*

```
CREATE TRIGGER  inc-import INSERT ON Càrrec
REFERENCING new AS nou
FOR EACH ROW
   (UPDATE Rebut r SET r.import = r.import + nou.import
   WHERE nou.dniSoci = r.dniSoci AND nou.mes = r.mes)
```

```
CREATE TRIGGER   dec-import  DELETE ON Càrrec
REFERENCING old AS vell
FOR EACH ROW
   (UPDATE Rebut r SET r.import = r.import - vell.import
   WHERE vell.dniSoci = r.dniSoci AND vell.mes = r.mes)
```

  - *Alguns SGBD poden permetre materialitzar la informació directament*

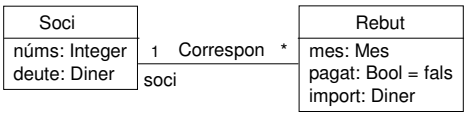
Tecnologia de les Bases de Dades Relacionals

Disseny d'operacions, assignació de responsabilitats al SGBD

- Els llenguatges actuals d'especificació no permeten definir un comportament actiu de l'esquema conceptual.
- En canvi, els SGBD relacionals sí que ho permeten.
- Per tant, en alguns casos els contractes d'especificació defineixen aspectes que el SGBD pot gestionar directament.

A disseny es pot assignar aquesta responsabilitat al SGBD
(v. Tema 4)

Tecnologia de les Bases de Dades Relacionals
Disseny d'operacions, assignació de responsabilitats al SGBD, exemple



Soci (dni, númSoci, deute)

Rebut (soci, mes, pagat, import)

context CapaDeDomini::BaixaSoci(dniSoci: Integer)
exc soci-no-existeix: no existeix un soci amb dniSoci
post 2.1: S'esborra el soci dniSoci
2.2 - S'esborren tots els rebuts del soci

Es passa al SGBD

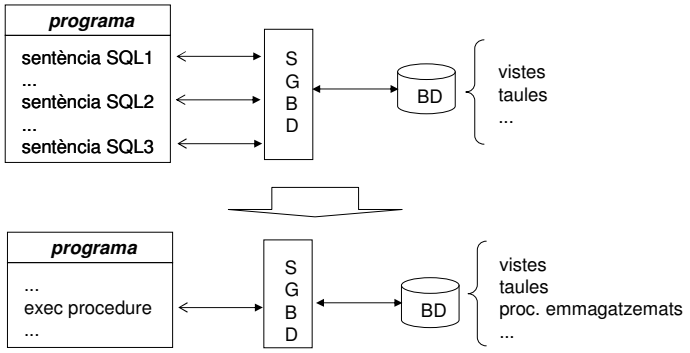
CREATE TABLE Soci (
dni STRING PRIMARY KEY,
númSoci INTEGER,
deute IMPORT,
UNIQUE númSoci)

CREATE TABLE Rebut (
soci STRING,
mes MES,
pagat BOOLEAN,
import IMPORT,
PRIMARY KEY (soci, mes),
FOREIGN KEY (soci) references(Soci) ON DELETE CASCADE)

En esborrar soci, el SGBD esborra automàticament els seus rebuts

Tecnologia de les Bases de Dades Relacionals
Disseny d'operacions, procediments emmagatzemats

- Serveixen per:
  - Simplificar el desenvolupament d'aplicacions
  - Millorar el rendiment de la base de dades
  - Controlar les operacions que els usuaris realitzen contra la base de dades
- Poden comprometre la portabilitat



Tecnologia de les Bases de Dades Relacionals
Disseny d'operacions, procediments emmagatzemats

Soci (dni, númSoci, deute, núm-cc)
Rebut (dniSoci, mes, pagat, import)
CompteCorrent (núm-cc, saldo)

```
CREATE PROCEDURE PagarRebut (dniS, mesEmissió)
RETURNING INTEGER, CHAR(50);
DEFINE       codi-error INTEGER; ….
ON EXCEPTION SET codi-error, miss-error;  RETURN codi-error, miss-errror END EXCEPTION;

IF ((SELECT COUNT(*) FROM rebut WHERE mes=mesEmissió AND dniSoci=dniS)=1) THEN
  LET import,l-pagat = (SELECT import, pagat FROM rebut  WHERE mes=mesEmissió AND dniSoci=dniS));
   IF ('Y' = l-pagat ) THEN RAISE EXCEPTION 2, 'El rebut ja està pagat';
      ELIF LET saldo, núm-cc = (SELECT c.saldo,c.núm-cc FROM soci s, comptecorrent c WHERE s.núm-cc=c.núm-cc
                                                                                    and s.dni=dniS);
            IF saldo < import THEN RAISE EXCEPTION 3, 'El soci no té prou saldo';
            ELSE UPDATE rebut SET pagat = 'Y' WHERE mes=mesEmissió AND dniSoci=dniS;
                  UPDATE soci s SET s.deute = s.deute-import WHERE númSoci = s.dni;
                  UPDATE comptecorrent c SET c.import = c.import-import WHERE núm-cc =c.núm-cc ENDIF
ELSE RAISE EXCEPTION 1, 'El soci no té rebut aquest mes'; END IF;
RETURN 0,'Rebut Pagat';
END PROCEDURE;
```


Estratègies de gestió de la persistència
Generació automàtica

- Generador automàtic de la persistència:
  - Sistema que proporciona una traducció automàtica per emmagatzemar les dades en memòria secundària
  - Transforma les actualitzacions fetes a memòria principal en actualitzacions en memòria secundària
  - Requereix especificar la correspondència entre el diagrama de classes (ja normalitzat) i l'esquema de la base de dades
- Característiques:
  - La gestió de la persistència és transparent al dissenyador
  - Pot no aprofitar-se completament de les funcionalitats ofertes pels SGBD

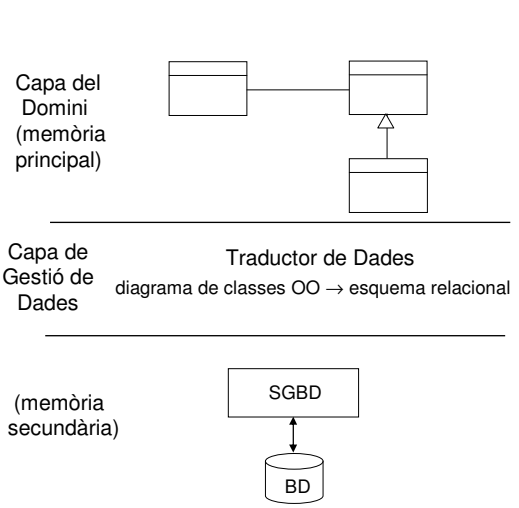
S'utilitza en conjunció amb el patró Domain Model a la capa de domini (tal i com hem fet durant el Tema 5)

Estratègies de gestió de la persistència
Disseny directe

- Disseny directe de la persistència:
  - El disseny del software es fa tenint en compte el SGBD que s'utilitzarà a la implementació
  - Serà diferent segons usem un SGBD orientat a objectes, relacional, etc.
- Característiques:
  - El dissenyador necessita conèixer el SGBD per dissenyar la capa de gestió de dades (i el sistema software en general)
  - Pot aprofitar-se completament de les funcionalitats ofertes pels SGBD
  - El disseny que es fa és similar per tots els sistemes concrets d'una mateixa família

S'utilitza en conjunció amb el patró Transaction Script a la capa de domini

Generació automàtica de la persistència
Patró Traductor de Dades (*Data Mapper*)

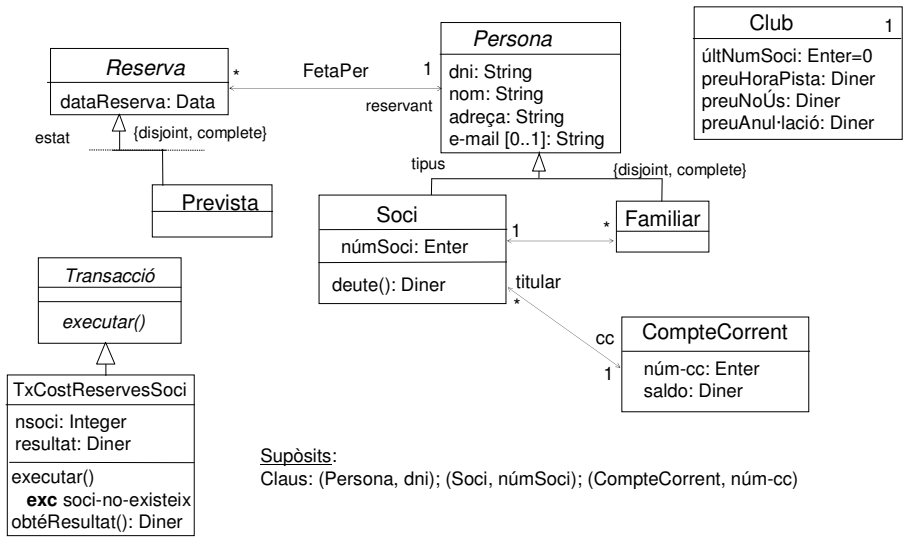


- El dissenyador defineix la correspondència entre el diagrama de classes normalitzat i l'esquema relacional segons les regles conegudes i d'altres possibles (p.e., tractament de patró Estat).
- El Traductor de Dades manté la correspondència entre els objectes i les taules.
- El programa Traductor de Dades s'assabenta (gairebé) automàticament de les modificacions fetes als objectes i les propaga a les taules.
- A l'etapa de disseny, els diagrames de seqüència no canvien
- Diferents productes implementen el patró de manera diversa:
  - Hibernate, TopLink, ...

Generació automàtica de la persistència
Operacions arbitràries de consulta a la capa de gestió de dades

- La generació automàtica de persistència és molt còmode per l'actualització de la BD però no tant per certes consultes
  - Diagrames de seqüència farragosos
  - Possible pèrdua d'eficiència
- Opció: la capa de dades ofereix les operacions de consulta requerides pel problema
  - La capa de domini fa de simple intermediària
  - Les consultes són arbitràriament complexes, i poden involucrar diverses taules
- Les opcions tecnològiques actuals de generació automàtica acostumen a oferir aquesta possibilitat:
  - Ex: Hibernate, *createQuery*

Generació automàtica de la persistència
Operacions arbitràries de consulta a la capa de gestió de dades, exemple

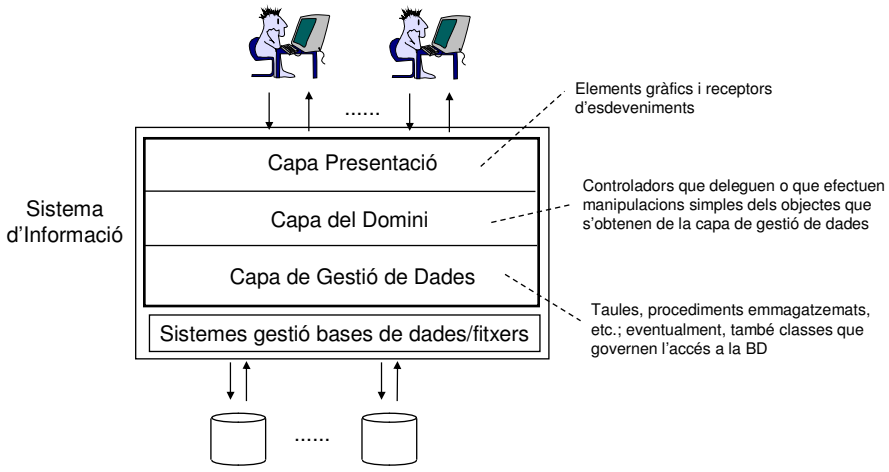


Supòsits:
Claus: (Persona, dni); (Soci, númSoci); (CompteCorrent, núm-cc)

Generació automàtica de la persistència
Operacions arbitràries de consulta a la capa de gestió de dades, exemple

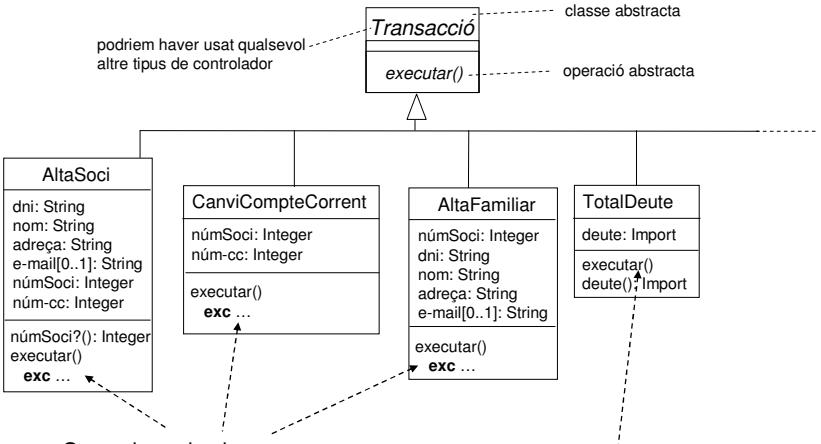
CtrlSoci 1
obté(dni: String): Soci **exc** soci-no-existeix obté(númSoci: Enter): Soci **exc** soci-no-existeix existeix(dni: String): Bool existeix(númSoci: Enter): Bool tots(): Set(Soci) costReservasSoci(ns: Enter): Diner **exc** soci-no-existeix …

Disseny directe de la persistència
Configuració de les capes



Disseny directe de la persistència

Capa de Domini: Disseny de les operacions



Operacions simples.
No s'assignen responsabilitats als objectes del diagrama (soci, familiar, etc.)
Accedeixen a la base de dades mitjançant crides a altres operacions

Disseny directe de la persistència

Disseny de les operacions, estratègies

- Controlador de la capa de domini amb lògica simple
  - El controlador crida operacions que accedeixen a la BD per obtenir dades i actualitzar-les
    - ✓ Ús de patrons d'accés a dades
  - El controlador fa manipulacions simples sobre aquestes dades
- Controlador de la capa de domini com a intermediari
  - El controlador delega en un procediment emmagatzemat que s'encarrega de tota la feina

És possible combinar les dues estratègies en un mateix sistema

En qualsevol dels dos casos, també es poden usar els elements de SQL vistos per assignar algunes de les responsabilitats

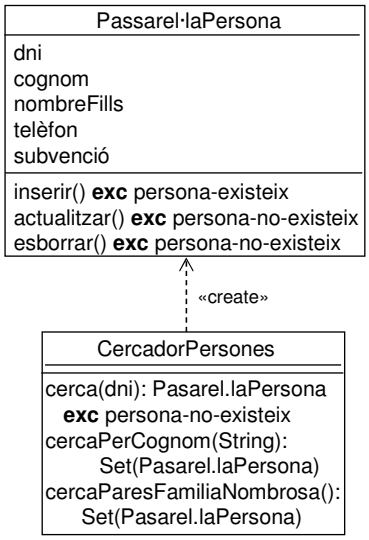
Disseny directe de la persistència

Controlador de la capa de domini amb lògica simple
Patrons de la capa de dades per al disseny directe en BD relacionals

- Passarel·la Taula (*Table Data Gateway*)
  - Una classe (d'una sola instància) per cada taula de la base de dades.
  - Encapsula l'accés (cerca, actualització) a una taula.
- Passarel·la Fila (*Row Data Gateway*)
  - Una classe per una taula de la base de dades. Una instància és una fila de la taula.
  - Les cerques poden ser operacions de classe o estar en una altra classe.
  - Encapsula l'actualització d'una fila.
  - Aplicable a models de disseny simples.
- Enregistrament actiu (*Active Record*)
  - Passarel·la fila amb operacions pròpies del domini.

Disseny directe de la persistència

Patró Pasarel.la Fila

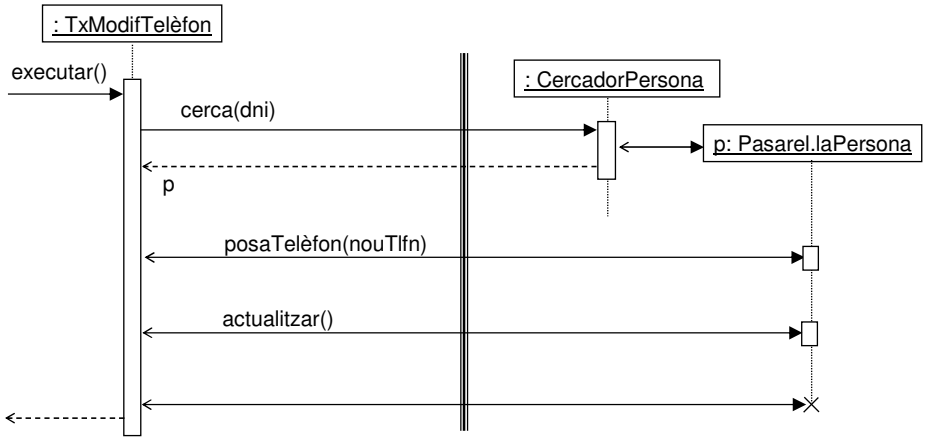


- Una classe passarel·la per taula (o vista). Les instàncies són files de la taula. Cada columna de la taula és un atribut de la classe.
- Les operacions de cerca consulten la taula corresponent mitjançant sentències SQL (o procediments emmagatzemats) i creen les instàncies de la passarel·la amb el resultat.
- Les operacions *inserir*, *actualitzar* i *esborrar* també contenen sentències SQL (o procediments emmagatzemats).

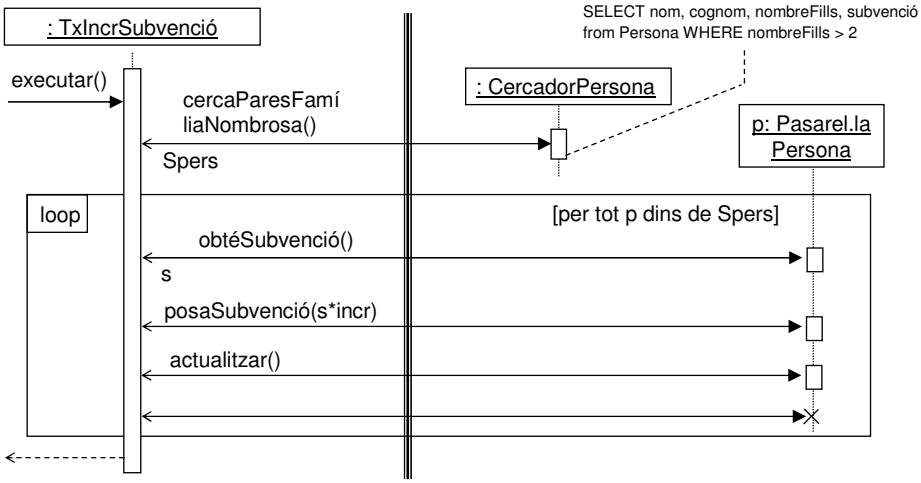
Disseny directe de la persistència
Disseny usant el patró Pasarel.la Fila

- El disseny de les operacions pot **aprofitar** la potència del **SQL**
- Els controladors de la capa de domini:
  - Creen pasarel.les per obtenir objectes
  - Manipulen aquests objectes a memòria principal
  - Els tornen a la capa de gestió de dades usant la pasarel.la

Disseny directe de la persistència
Disseny usant el patró Pasarel.la Fila, exemple



Disseny directe de la persistència
Disseny usant el patró Pasarel.la Fila, un altre exemple

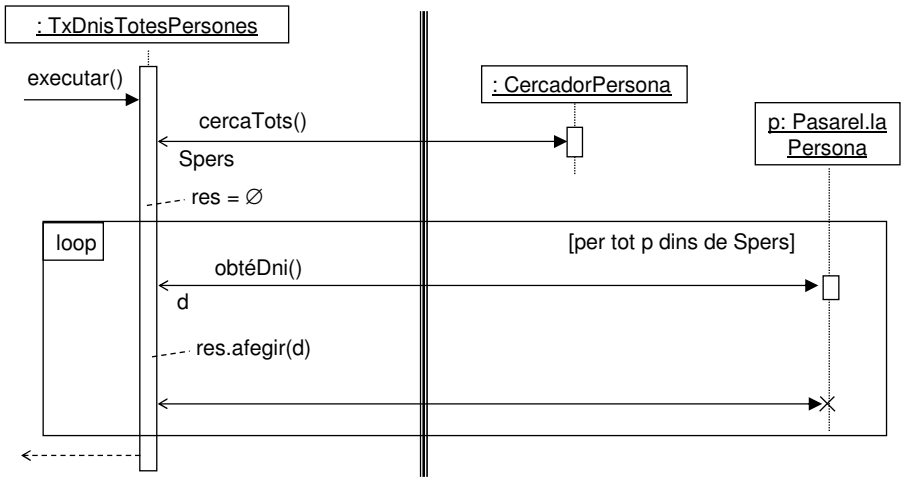


Disseny directe de la persistència
Disseny usant el patró Pasarel.la fila amb operacions arbitràries de consulta

- Igual que passava amb Domain Model, algunes operacions de consulta poden ser feixugues usant el patró estrictament
- Solució: permetre operacions de consulta en els cercadors
  - Obtenció de col.leccions
  - Càlcul de resultats
- Aquestes operacions arbitràries de consulta permeten consultar diverses taules de la base de dades, i retornar valors de qualsevol tipus

Disseny directe de la persistència

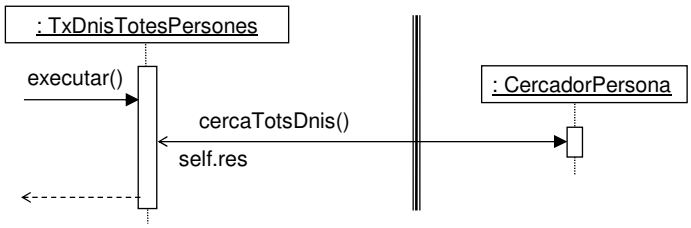
Disseny usant el patró Pasarel.la Fila, exemple amb operacions de consulta simples



context CercadorPersona::cercaTots(): Set(Pasarel.laPersona)

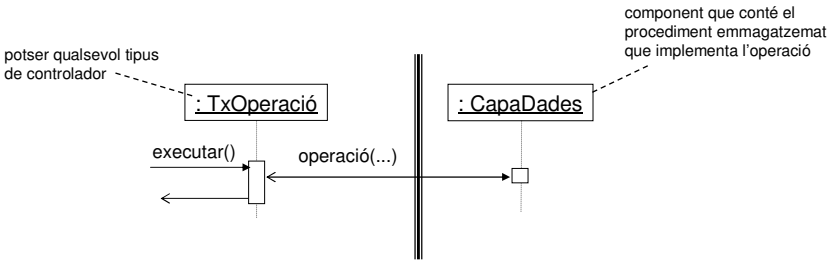
Disseny directe de la persistència

Disseny usant el patró Pasarel.la Fila, exemple amb operacions arbitràries de consulta



context CercadorPersona::cercaTotsDnis(): Set(String)

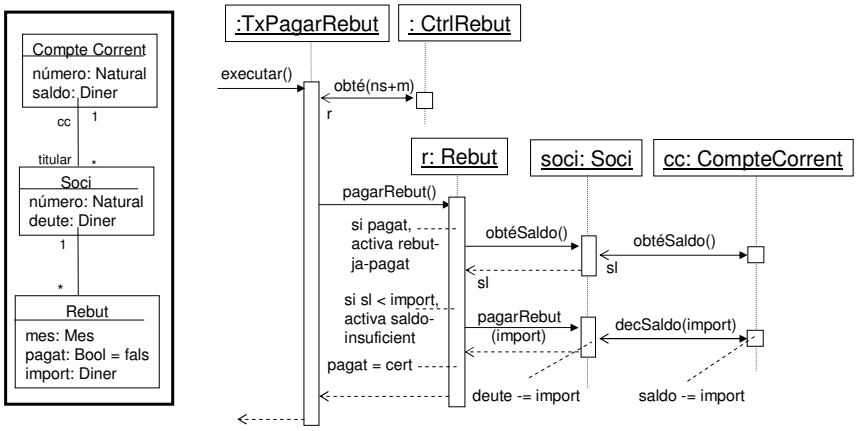
Disseny directe de la persistència
Controlador de la capa de domini com a simple intermediari



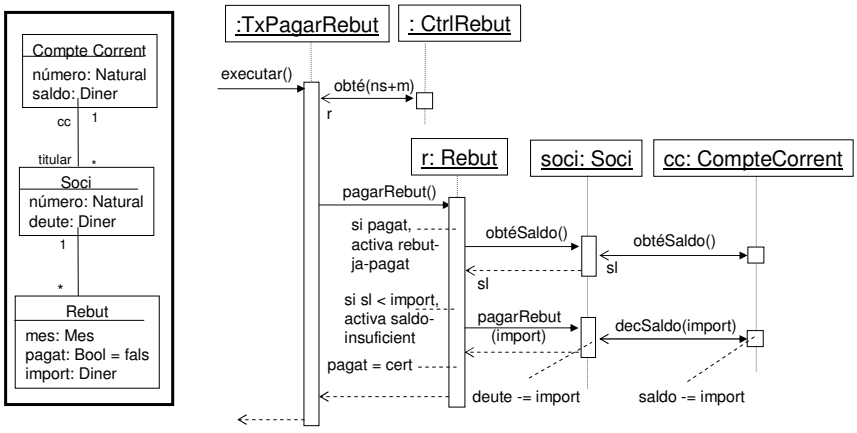
Exemple de gestió de la persistència
Enunciat

context pagarRebut (ns: Enter, m: Mes)
-- operació per pagar el rebut del mes *m* del Soci *ns*
exc rebut-no-existeix**:** no existeix un rebut del soci *ns* el mes *n*
rebut-ja-pagat: el rebut ja està pagat
saldo-insuficient: el client no té prou saldo al seu compte
post:
paga-rebut: l'atribut *pagat* de *Rebut* es posa a cert
decrementa-saldo: el saldo del compte corrent del soci es decrementa en l'import del rebut
decrementa-deuta: el deute del soci es decrementa en l'import del rebut

Exemple de gestió de la persistència
Solució sense considerar persistència

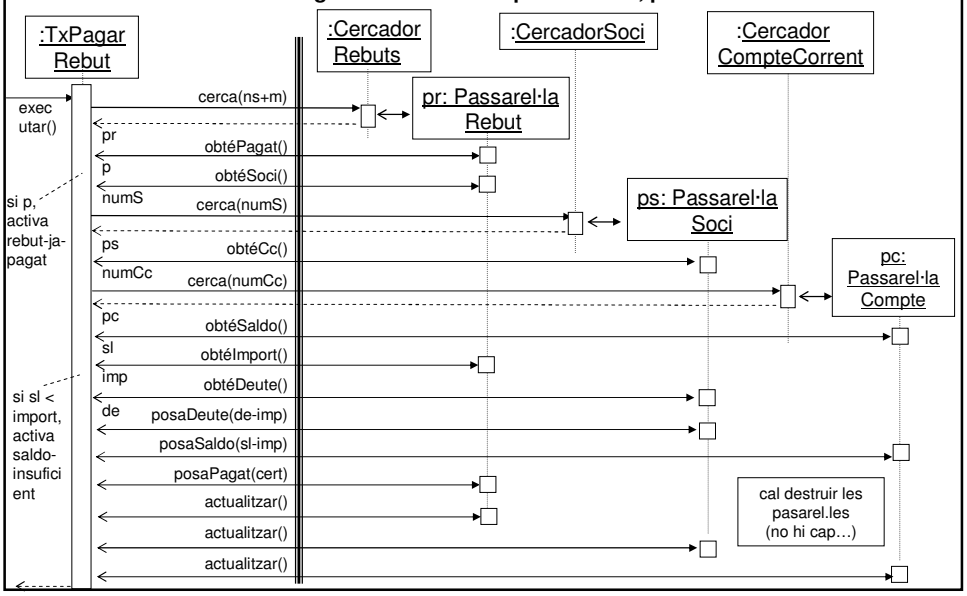


Exemple de gestió de la persistència
Solució amb generació automàtica de persistència



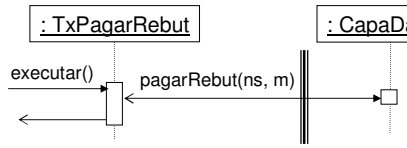
Exemple de gestió de la persistència

Solució amb gestió directe de la persistència, pasarel.la fila



Exemple de gestió de la persistència

Solució amb gestió directe de la persistència, procediment emmagatzemat



Soci (dni, númSoci, deute, núm-cc)
Rebut (soci, mes, pagat, import)
CompteCorrent (núm-cc, saldo)

```
CREATE PROCEDURE pagarRebut (númS, mesEmissió)
RETURNING INTEGER, CHAR(50);
DEFINE       codi-error INTEGER; ….
ON EXCEPTION SET codi-error, miss-error;  RETURN codi-error, miss-error END EXCEPTION;

IF ((SELECT COUNT(*) FROM rebut WHERE mes=mesEmissió AND núms=númSoci)=1) THEN
  LET import,l-pagat = (SELECT import, pagat FROM rebut  WHERE mes=mesEmissió AND núms=númSoci));
    IF ('Y' = l-pagat ) THEN RAISE EXCEPTION 2, 'El rebut ja està pagat';
       ELIF LET saldo, numcc = (SELECT c.saldo,c.núm-cc FROM soci s, comptecorrent c WHERE s.núm-cc=c.núm-cc
                                                                                    and s.númSoci=númS);
            IF saldo < import THEN RAISE EXCEPTION 3, 'El soci no té prou saldo';
            ELSE UPDATE rebut SET pagat = 'Y' WHERE mes=mesEmissió AND núms=númSoci;
                 UPDATE soci s SET s.deute = s.deute-import WHERE númS = s.númSoci;
                 UPDATE comptecorrent c SET c.import = c.import-import WHERE núm-cc =c.núm-cc ENDIF
ELSE RAISE EXCEPTION 1, 'El soci no té rebut aquest mes'; END IF;
RETURN 0,'Rebut Pagat';
END PROCEDURE;
```

Disseny de les capes de domini i gestió de dades, resum

- Els patrons de les capes de domini i de gestió de dades estan relacionats:
  - Domain Model + Data Mapper
    - variant: Domain Model amb consultes a la capa de dades
  - Transaction Script: permet dues variants
    - Transaction Script + Pasarel.la fila
      - Variant: Pasarel.la fila amb consultes arbitràries
    - Transaction Script + Procediment emmagatzemats
    - Poden combinar-se en el mateix sistema
- En qualsevol opció, es pot implementar directament en l'esquema de la BD les responsabilitats assignades a la capa de gestió de dades en fer l'assignació de responsabilitats a capes (v. Tema 4):
  - Provinents de restriccions d'integritat i informació derivada (e.g., claus)
  - Provinents dels contractes d'especificació (e.g., esborrat en cascada)
  - En el cas de Data Mapper, depèn del que la tecnologia emprada permeti fer

Bibliografia

- M. Fowler
  Patterns of Enterprise Application Architecture
  Addison-Wesley, 2003
- H. Garcia-Molina, J. Ullman, J. Widom
  Database Systems Implementation
  Prentice-Hall, 2000.
- *Understanding SQL and Java Together*
  J.Melton, A.Eisenberg
  Morgan-Kaufmann, 2000.
- *Hibernate in Action*
  Christian Bauer, Gavin King
  Manning Publications Company, 2004