

Approche « supply driven » : focus détaillé

Giuseppe Berio

giuseppe.berio@univ-ubs.fr

2023



Focus détaillé

- ▶ Transformation de schémas
- ▶ Échange de données
- ▶ Intégration de schéma
- ▶ Intégration de données

Approche « supply-driven » : perspective méthodes/techniques

3

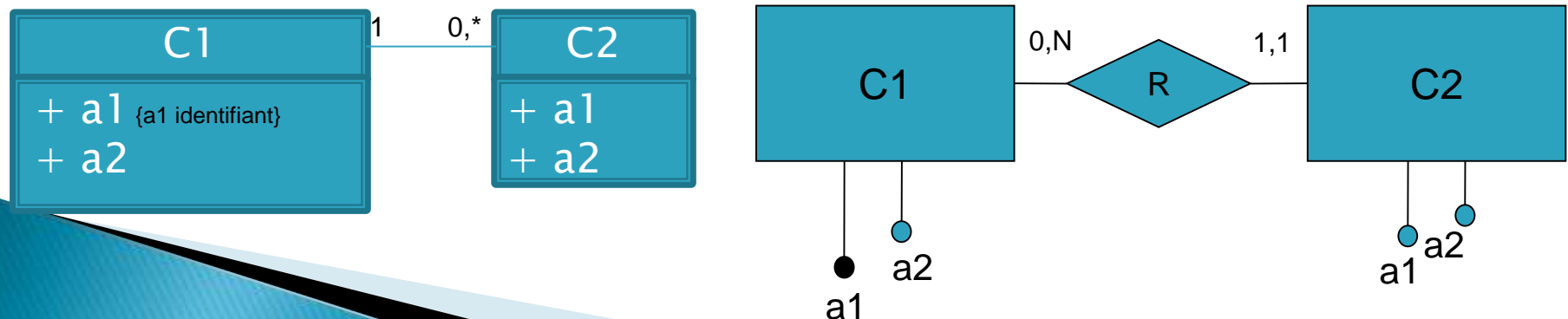
Identification sources	Conception SI	Conception SD	Conception Flux	Conception
Analyse (statique) de qualité : choix de sources	Transf Schéma ($C \rightarrow C$)	Transf Schéma ($C \rightarrow L; C \rightarrow C; C \rightarrow L$)	Échange de données : Mappings (conceptuel)	
Transf Schéma ($L \rightarrow C; C \rightarrow L$)	Intégration schéma (conceptuel) + Transf Schéma ($C \rightarrow L$)		Intégration de données (conceptuel)	Mise en œuvre
Réimplantation sources	Implantation SI	Implantation SD	Programmation flux	
Transf Schéma ($L \rightarrow P$ sources) + déploiement	Transf Schéma : $L \rightarrow P$ schéma intégré) + déploiement	Transf Schéma : $L \rightarrow P, L \rightarrow L$ schéma dimensionnel) + déploiement	Flux de chargement (SI-SD) + déploiement	
			Flux d'intégration (complétude + non redondance) (Sources-SI) + déploiement	
			Flux d'extraction et d'amélioration de la qualité (Source-Source) + déploiement	

Transformation des schémas (TS)

- ▶ Définition : produire un **schéma cible** à partir d'un **schéma source**
- ▶ Objectifs : **points de vue**, évolution et migration, implantation, rétro-ingénierie
- ▶ **Notion d'équivalence ?**
 - N'est pas une nécessité formelle, même si elle peut avoir un intérêt (**traduction des schémas**, comme par exemple dans le cas de la normalisation relationnelle) car, si présentes, les données seront adaptées en fonction de la transformation (échange de données)
 - Souvent basée sur les données source qui doivent pouvoir être reconstruites à partir de données organisées selon le schéma cible (données source = F(données cibles))
- ▶ **Aspect critique** : la transformation doit être **justifiée** et **une liaison entre le schéma origine et le schéma résultat** doit être disponible (mais les données ne sont pas forcément en jeu dans cette liaison) ; le schéma résultat doit être testé (pour complétude et correction)

Transformation de schéma : Le langage commun

- ▶ Le langage commun est un **choix contextuel**
- ▶ Ce langage peut être un langage pour des **modèles logiques** de données (par exemple, le relationnel) et/ou pour des **modèles conceptuels** (par exemple, MERISE/EA/UML)
- ▶ Ce choix est typiquement un langage pour modèles conceptuels (car les autres sont généralement imposés) ; par exemple :



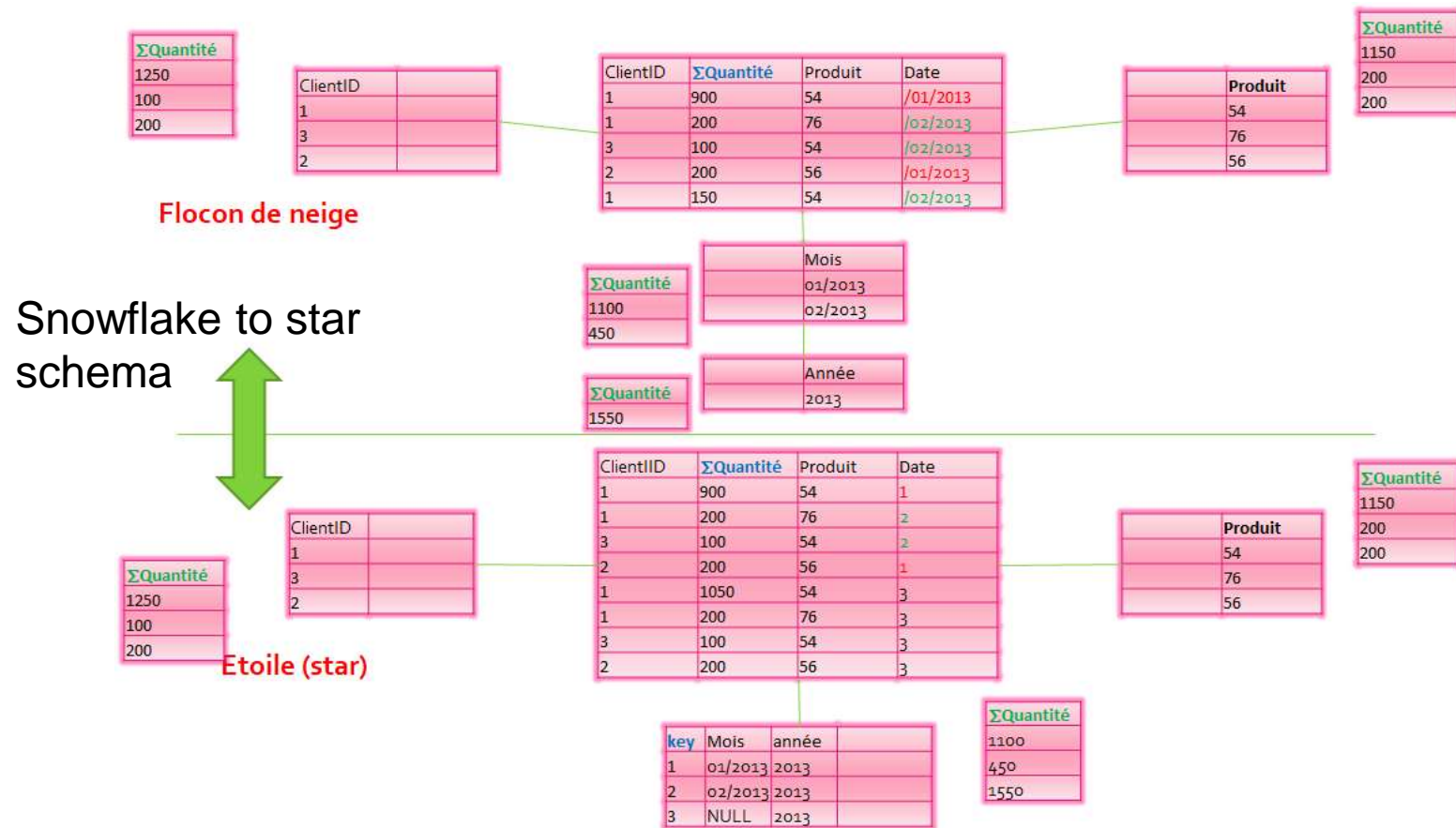
Types de transformation

- ▶ Types couramment utilisés :
 - Conceptual to logical (C to L) – exemple, mise en œuvre d'un schéma conceptuel via une bdd relationnelle transactionnelle
 - **Logical to logical (L to L)** – exemple, « normalisation relationnelle », « flocon → étoile », « colonnes/tables additionnelles »
 - **Logical to conceptual (L to C)** – exemple, « relationnel → diagramme ER »
 - **Conceptual to conceptual (C to C)** – exemple, « classes/attributs additionnels »
- ▶ En complément : Logical to Physical (L to P) et vice-versa (P to L)
- ▶ Ces types de transformation peuvent être composés, en particulier dans le cas des entrepôts
- ▶ Certains types sont « naturellement automatisables » : $P \rightarrow L, C \rightarrow L$; pour d'autres l'automatisation ne fournit qu'une base, parfois inutile

Transformation « conceptual to logical »

- ▶ Il s'agit de la transformation classique pour, par exemple, à partir d'un modèle plutôt conceptuel (comme un modèle UML de classes métiers ou un modèle ER et similaires) des tables relationnelles sont générées formant le schéma d'une base de données
- ▶ Ces transformations sont typiquement automatisables en grande partie

Transformation « logical to logical »



Transformation « logical to logical »

Professor (Id Name Sal)

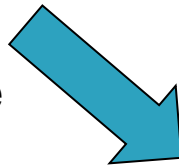
Clés primaires
surlignées

Student (Name Yr)

PayRate (Rank HrRate)

WorksOn (Name Proj Hrs ProjRank)

Abstraction (agrégation de
données, perte de détail)



Personnel (Id Name Sal Addr)

Transformation « logical to logical »

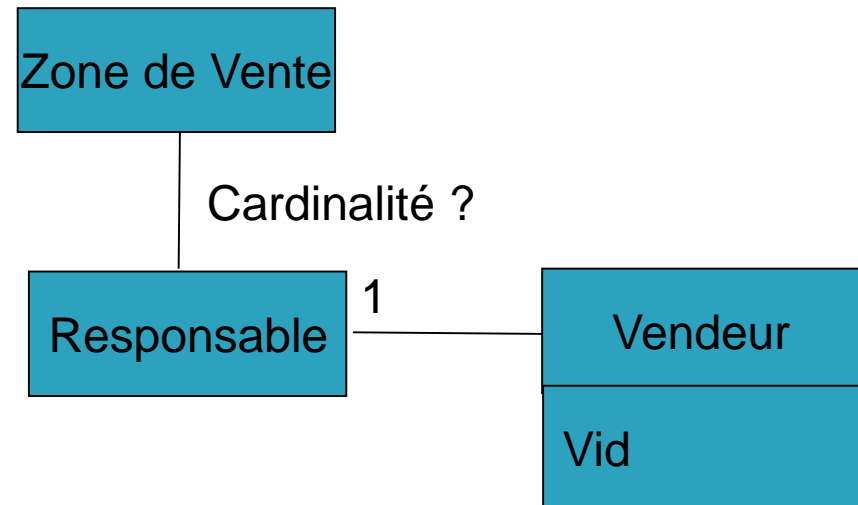
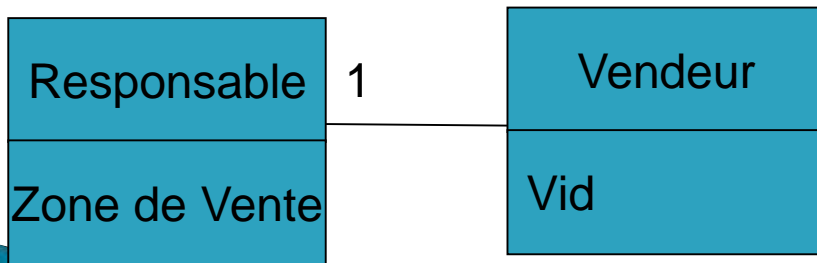
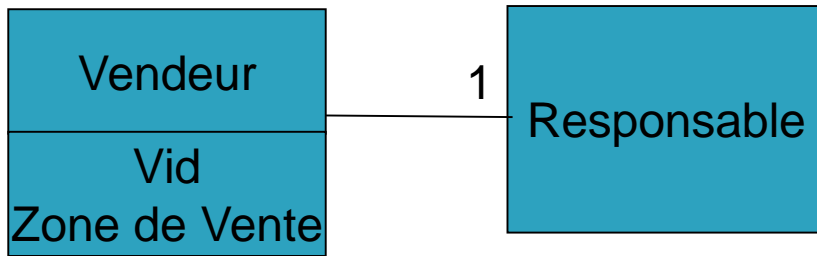
- ▶ (Re)Introduction de contraintes mono-table (check, not null, unique)
- ▶ Explicitation de valeurs calculées
- ▶ Adaptation du type

Transformation « logical to conceptual »

Vendeurs(Vid, Nom, Zone de Vente, Responsable)

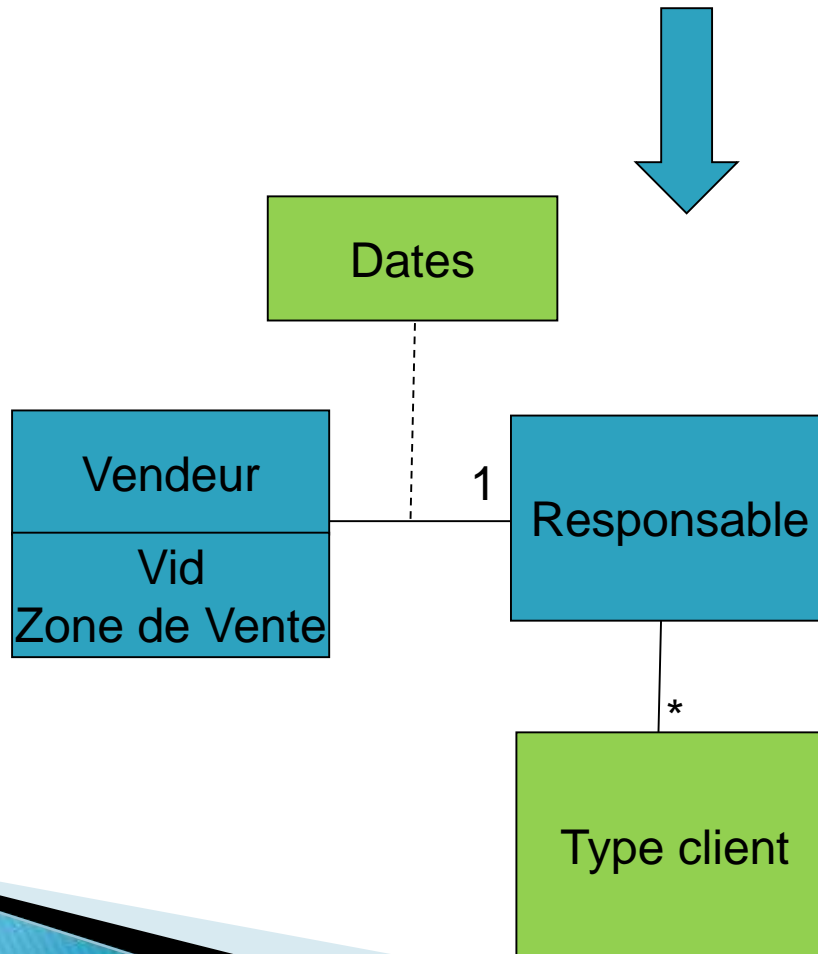


Transformations
possibles vers un
schéma UML_like
(enrichissement
semantique, distinction
relation/classe,
normalisation)



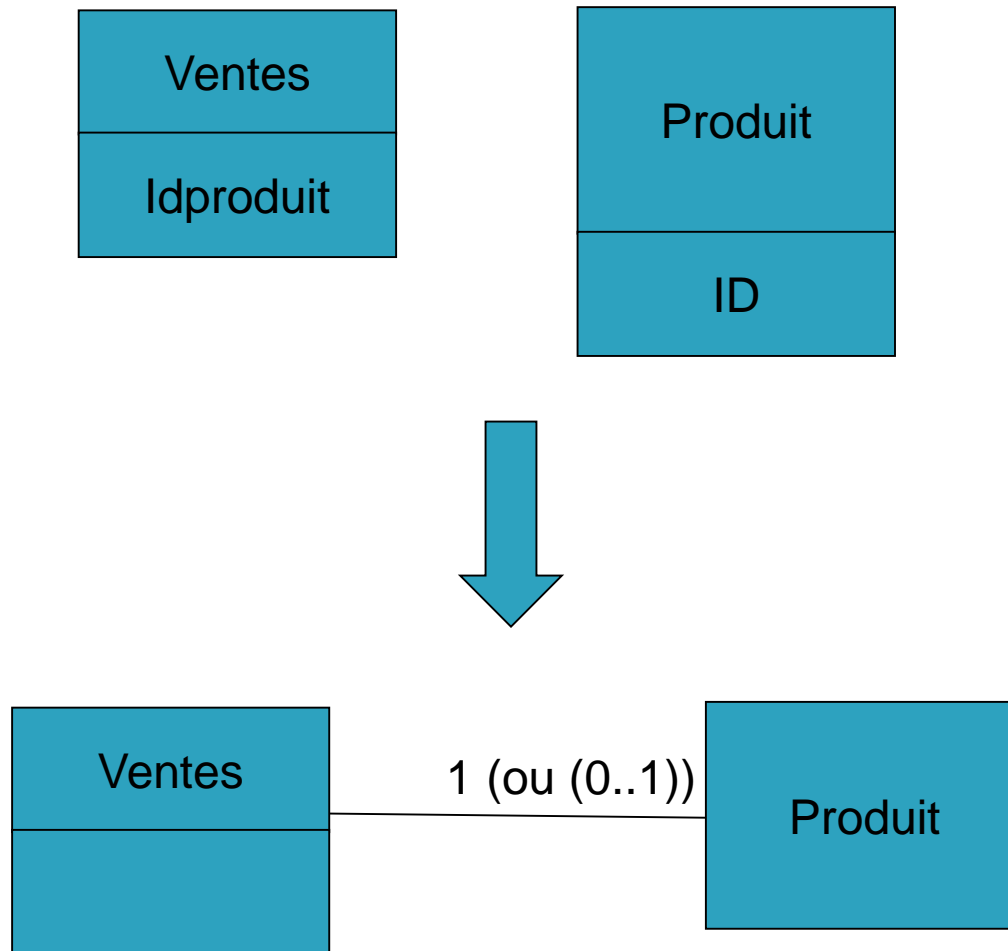
Transformation « logical to conceptual »

Vendeurs(Vid, Nom, Zone de Vente, Responsable)



Transformations possibles
vers un schéma UML-like
(enrichissement
semantique/nouvelles
données, même inconnues)

Transformation « logical to conceptual »

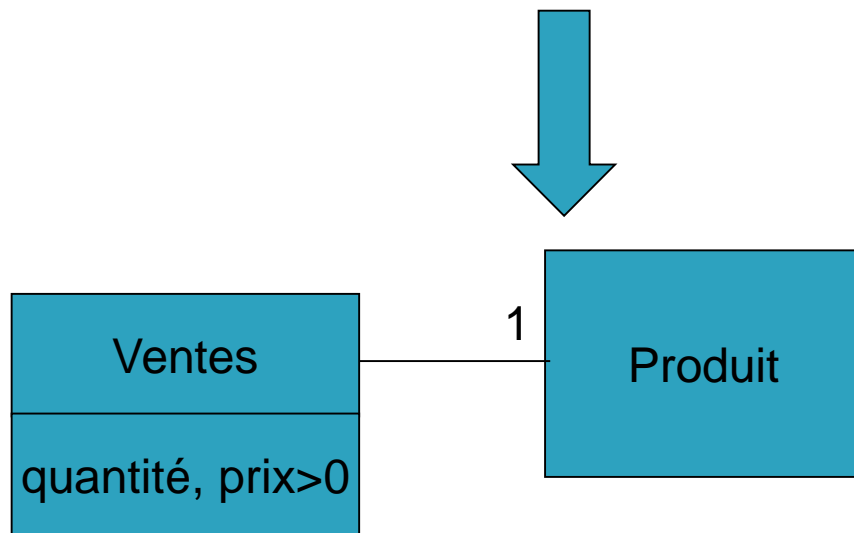


Enrichissement :
rajout des relations
(ou introduction de
clés étrangères)

Transformation « logical to conceptual »

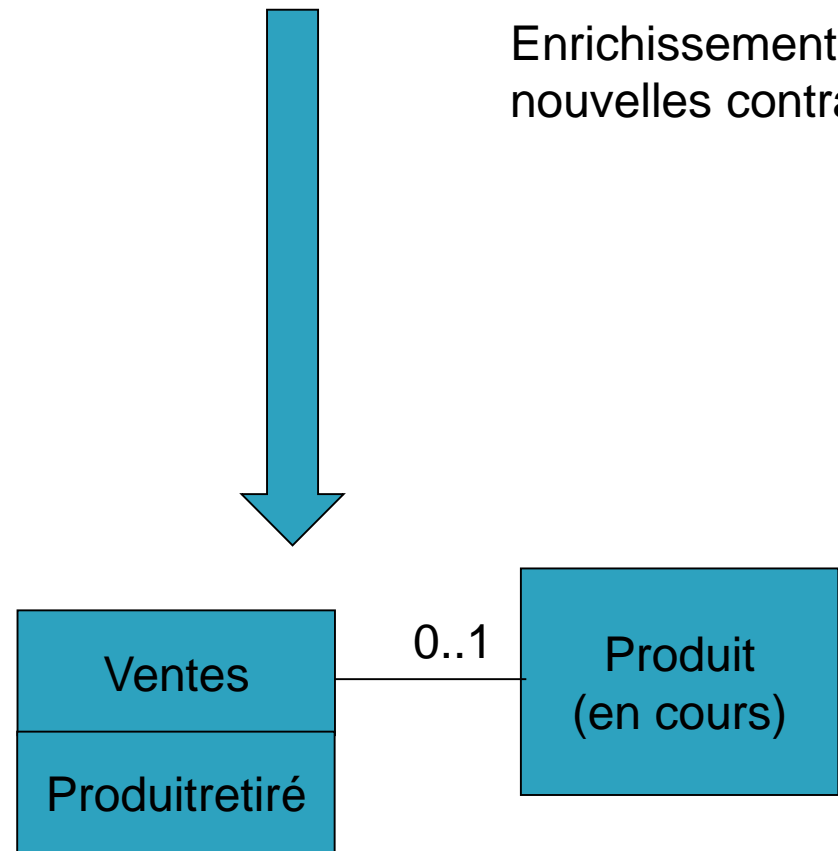
Produit(pid, prix, descriptif)

Ventes (produit not null, quantité, prix) sans clé étrangère sur Produit



(Ventes.Produit is not Null or
Produitretiré is not Null) and
(Vente.Produit is null or
Produitretiré is null)

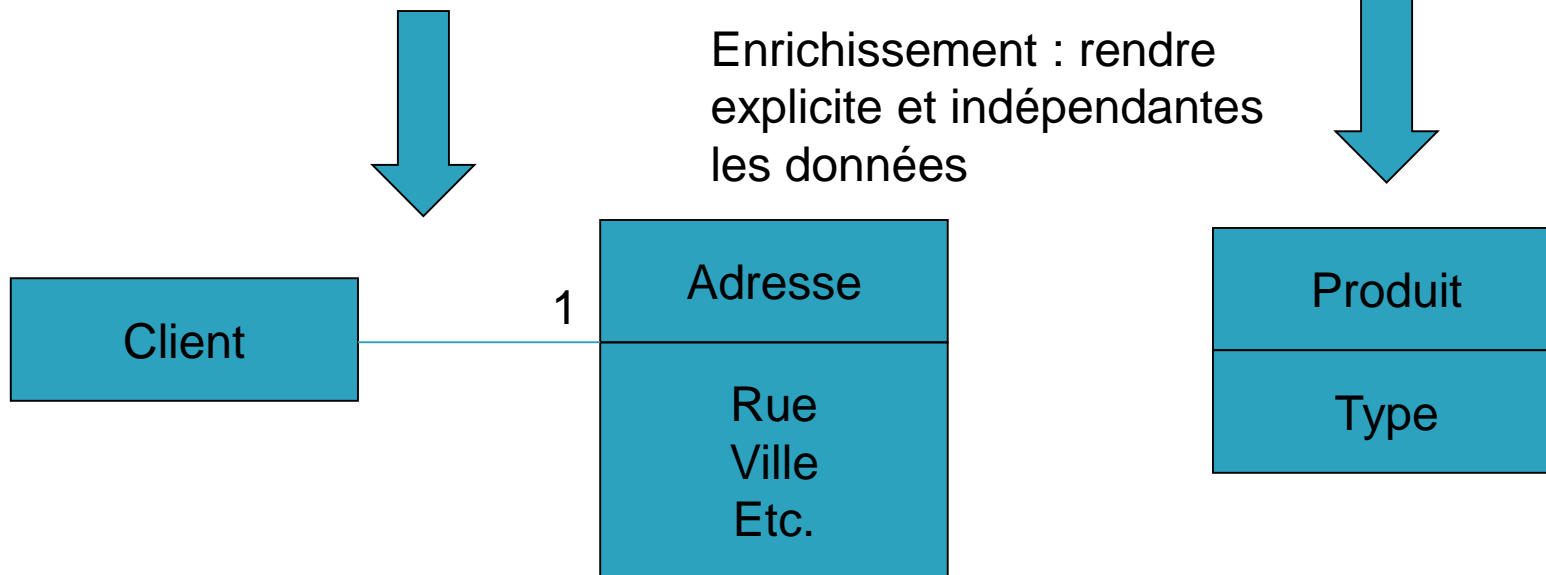
Enrichissement :
nouvelles contraintes



Transformation « logical to conceptual »


Client(cid, nom, prenom, adresse)
/* si l'on connaît le format de
«adresse» et on sait le traiter

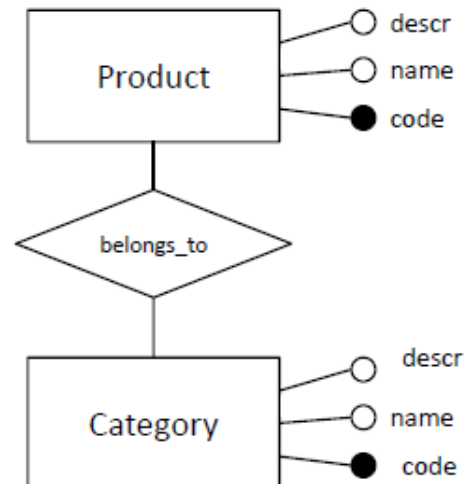
Produit(pid, descriptif, prix) /* si
l'on connaît le format de «pid» et
on sait le traiter, par exemple on
sait pouvoir extraire un «type» ou
un «type» est connu en fonction
de la source



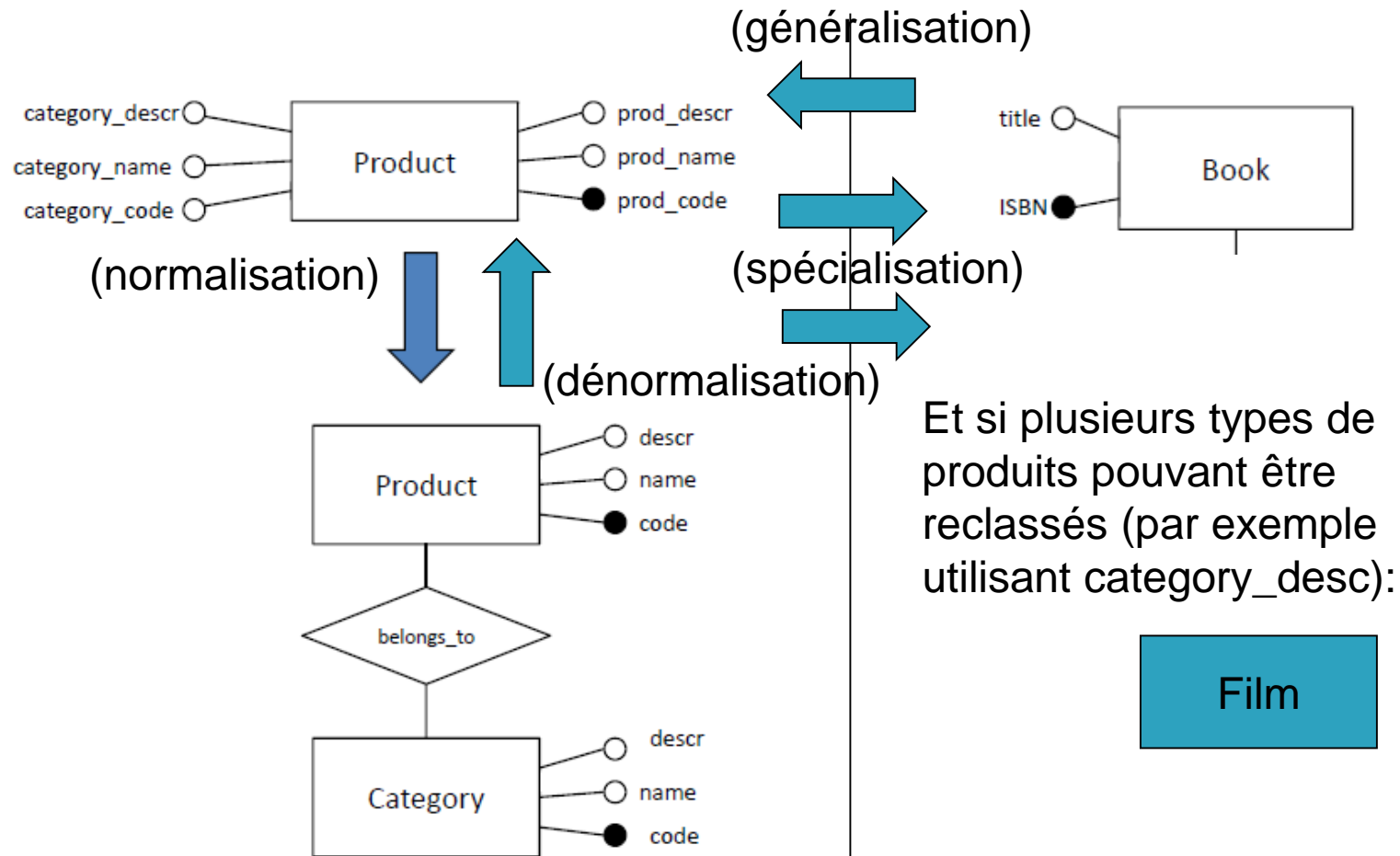
Transformation « conceptual to conceptual »



Transformation (C to C)  (normalisation : rendre indépendantes les données)



Transformation « conceptual to conceptual »

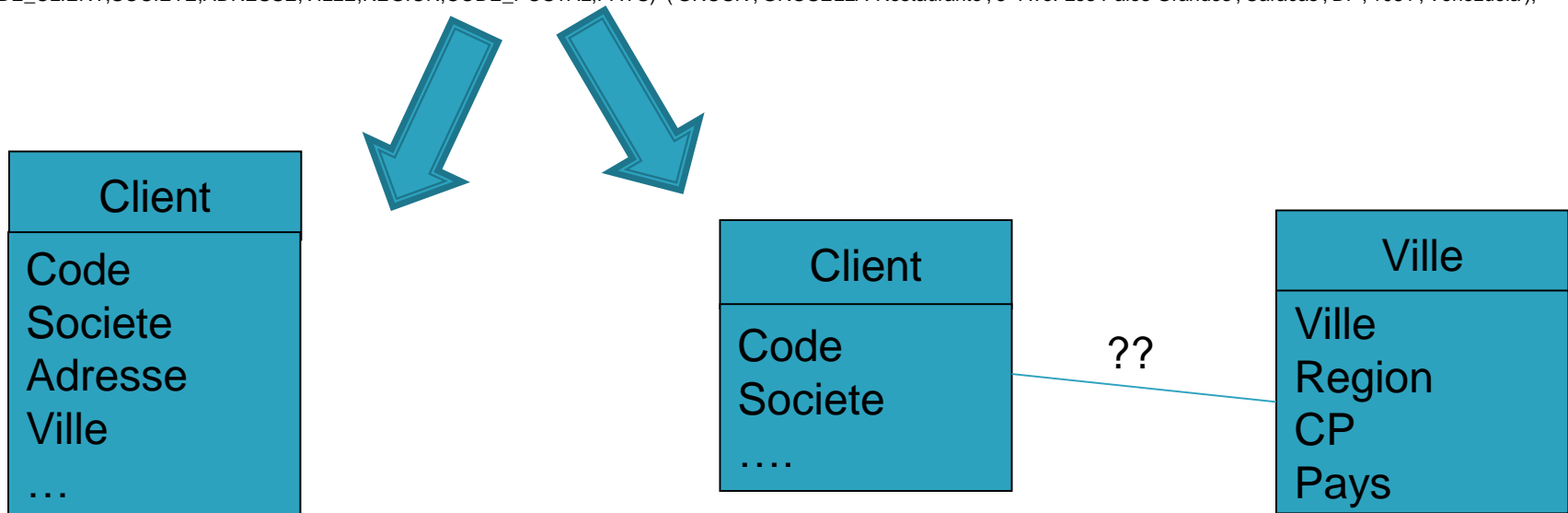


Transformation « logical to conceptual »

- ▶ Toute typologie de source (sgbd relationnels ou non relationnels – **noSQL**–, autres structures, fichiers textuels (**XML**, **CSV**,...), doit être modélisée d'une manière conceptuelle faisant ressortir
 - La signification de données plutôt que la forme
 - La notion d'identifiant au sens conceptuel, plutôt que d'identifiant au sens de l'implémentation informatique
- ▶ En général, cette tâche n'est pas trop complexe, d'autant plus que ces formats ont été extensivement explorés (voir documents sur la plateforme : ERXML, ERnoSQL, ERnoSQL2)

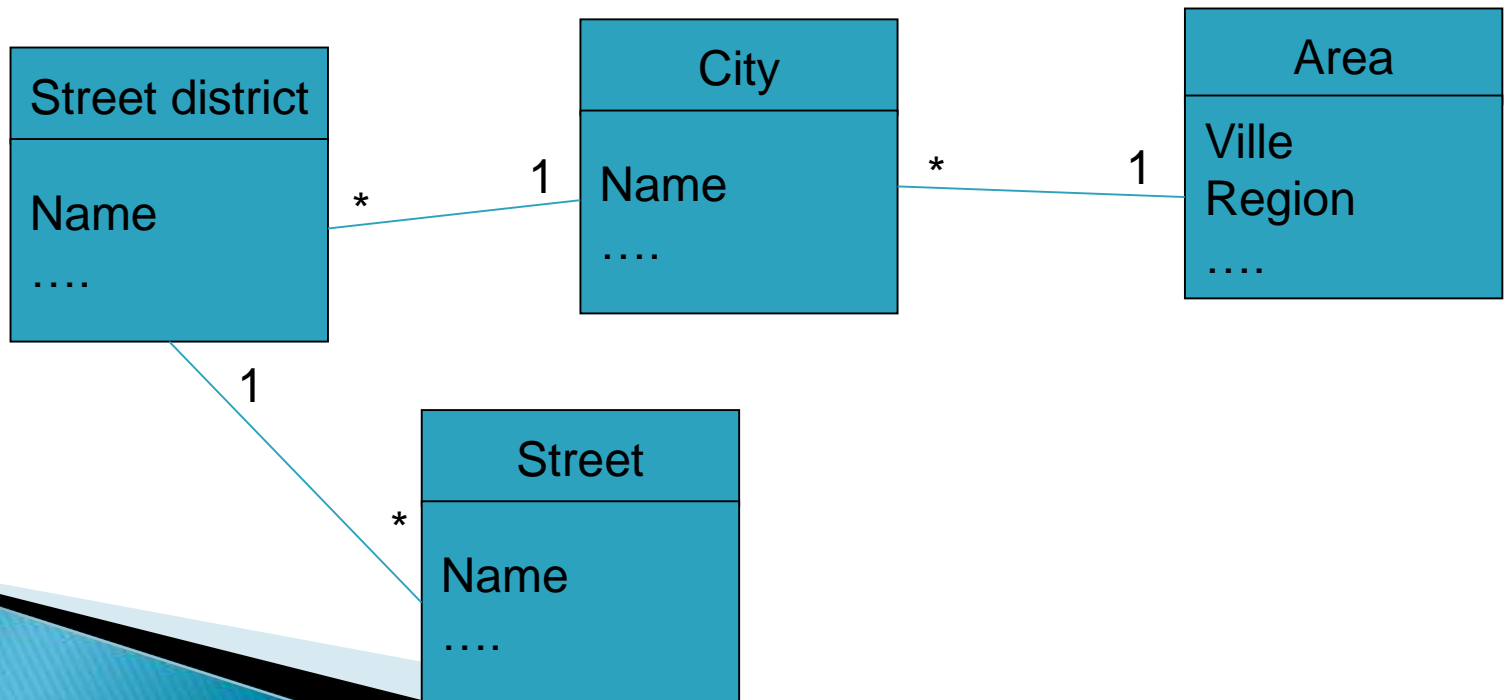
Exemple (txt)

(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('VINET','Vins et alcools Chevalier','59 rue de l'Abbaye','Reims',null,'51100','France');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('HANAR','Hanari Carnes','Rua do Paço, 67','Rio de Janeiro','RJ','05454-876','Brésil');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('VICTE','Victuailles en stock','2, rue du Commerce','Lyon',null,'69004','France');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('SUPRD','Suprêmes délices','Boulevard Tirou, 255','Charleroi',null,'B-6000','Belgique');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('CHOPS','Chop-suey Chinese','Hauptstr. 29','Bern',null,'3012','Suisse');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('HILAA','HILARIÓN-Abastos','Carrera 22 con Ave. Carlos Soublette #8-35','San Cristóbal','Táchira','5022','Venezuela');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('ERNSH','Ernst Handel','Kirchgasse 6','Graz',null,'8010','Autriche');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('OTTIK','Ottilies Käseladen','Mehrheimerstr. 369','Köln',null,'50739','Allemagne');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('QUEDE','Que Delícia','Rua da Panificadora, 12','Rio de Janeiro','RJ','02389-673','Brésil');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('RATTC','Rattlesnake Canyon Grocery','2817 Milton Dr.','Albuquerque','NM','87110','États-Unis');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('FOLKO','Folk och få HB','Åkergatan 24','Bräcke',null,'S-844 67','Suède');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('FRANK','Frankenversand','Berliner Platz 43','München',null,'80805','Allemagne');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('GROSR','GROSELLA-Restaurante','5ª Ave. Los Palos Grandes','Caracas','DF','1081','Venezuela');



Exemple (xml)

```
<areas>  
  <area city="Paris">  
    <street district="2eme arrondissement">Rue de la Paix</street>  
  </area>  
  <area city="Paris">  
    <street district="8eme arrondissement">Champs Elysees</street>  
  </area>  
  <area city="New York City">  
    <street district="Manhattan">Madison avenue</street>  
  </area>  
  <area city="New York City">  
    <street district="Brooklyn">Washington heights</street>  
  </area>  
</areas>
```



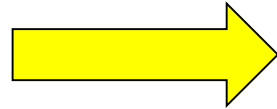
Transformation « logical to logical » par composition

Professor (Id Name Sal)

Student (Name Yr)

PayRate (Rank HrRate)

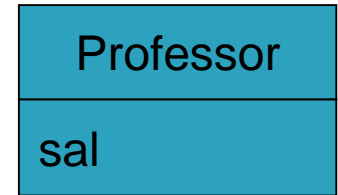
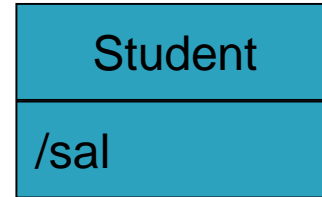
WorksOn (Name Proj Hrs ProjRank)



Personnel (Id Name Sal Addr)



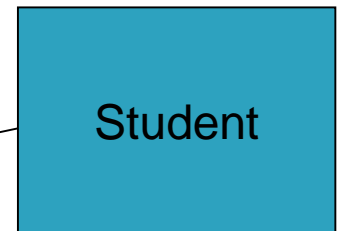
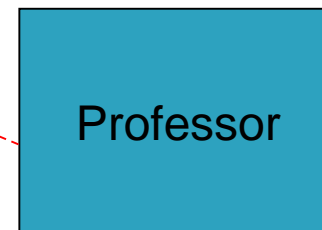
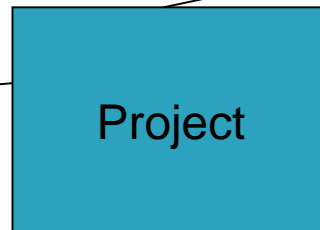
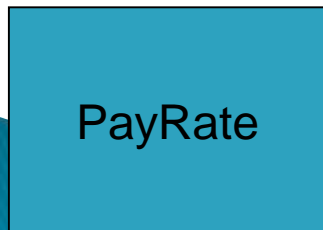
Transformation (C to L)



Transformation L to C
vers un schéma
UML_like (affiché sans
attributs)



Transformation (C to C)



1

*

1

*

??

Transformations de schémas dans l'approche « supply driven »

Identification sources	Conception SI	Conception SD	Conception Flux	Conception
Analyse (statique) de la qualité de données et choix de sources	Transf Schéma (C→C) : résoudre les conflits entre schémas distincts portants sur de données se référant à une même réalité	Transf Schéma (C→C) : obtenir un schéma multidimensionnel se basant sur les données et schéma intégrés	Conception de flux d'extraction : Échange de données visant la limitation de données erronées, la standardisation (mappings conceptuels)	
Transf Schéma (L→C) : expliciter les données, améliorer la qualité (au sens large) de données à transférer vers l'entrepôt	Intégration schéma (conceptuel) : produire de schéma à partir de plusieurs schémas, limitant (excluant) la redondance de données et augmentant leurs complétude		Conception de flux d'intégration : Intégration de données visant non redondance, complétude et fraîcheur (mappings conceptuels)	
Réimplantation sources	Implantation SI	Implantation SD	Programmation flux	Mise en œuvre
Transf Schéma (C→L;L→P sources) + déploiement	Transf Schéma : C→L;L→ P schéma intégré) + déploiement	Transf Schéma : L→ L, L→ P schéma dimensionnel)+ déploiement	Flux de chargement : Échange de données (chargements successifs)+ déploiement (parallélisation)	
			Flux d'intégration (intégrations successives) + déploiement (parallélisation)	
			Flux d'extraction : Échange de données (mappings exécutables et extractions successives)+ déploiement (parallélisation)	

Points clé :

- Les transformations en rouge sont complexes ; plus ces transformations sont soignées, plus l'intégration de schémas sera simple ;
- Les transformations en vert sont généralement plus simples car bien cadrées, quoique parfois il faut faire des choix et la transformation vers un schéma physique peut être assez articulée, en particulier dans les cas des entrepôts

Identification des sources :

Transformations de schémas ($L \rightarrow C$)

- ▶ Une transformation de schémas fournit des hypothèses de qualité sur les données disponibles
- ▶ Exemple (introduction d'une contrainte)
 - $\text{Prix} > 0$ (Hypothèse)
 - Lancement d'une requête (ou similaire) pour compter les données ayant un $\text{Prix} \leq 0$
 - S'il y en a pas (ou peu), la transformation peut être validée
 - S'il y en a, le choix est entre « données erronées » et « hypothèse non validée »
 - Si « données erronées » alors, validation de la transformation
 - Si « hypothèse non validée » alors la situation doit être approfondie

Identification des sources :

Transformations de schémas

► Exemple (normalisation)

- Dans une table, l'on suppose l'existence d'une dépendance fonctionnelle $A \rightarrow B$, A n'étant pas clé de la table (ou UID d'entité ou contrainte classe PK-like)
- Lancement d'une requête (ou similaire) pour compter (ou visualiser) les données ne respectant pas la dépendance fonctionnelle
 - S'il y en a pas (peu), la transformation peut être validée (rajout contrainte « unique » ou décomposition de table/entité/classe)
 - S'il y en a, le choix est entre « données erronées » et « hypothèse non validée »
 - Si « données erronées » alors validation de la transformation
 - Si « hypothèse non validée » alors la situation doit être approfondie

Identification des sources : Transformations de schémas

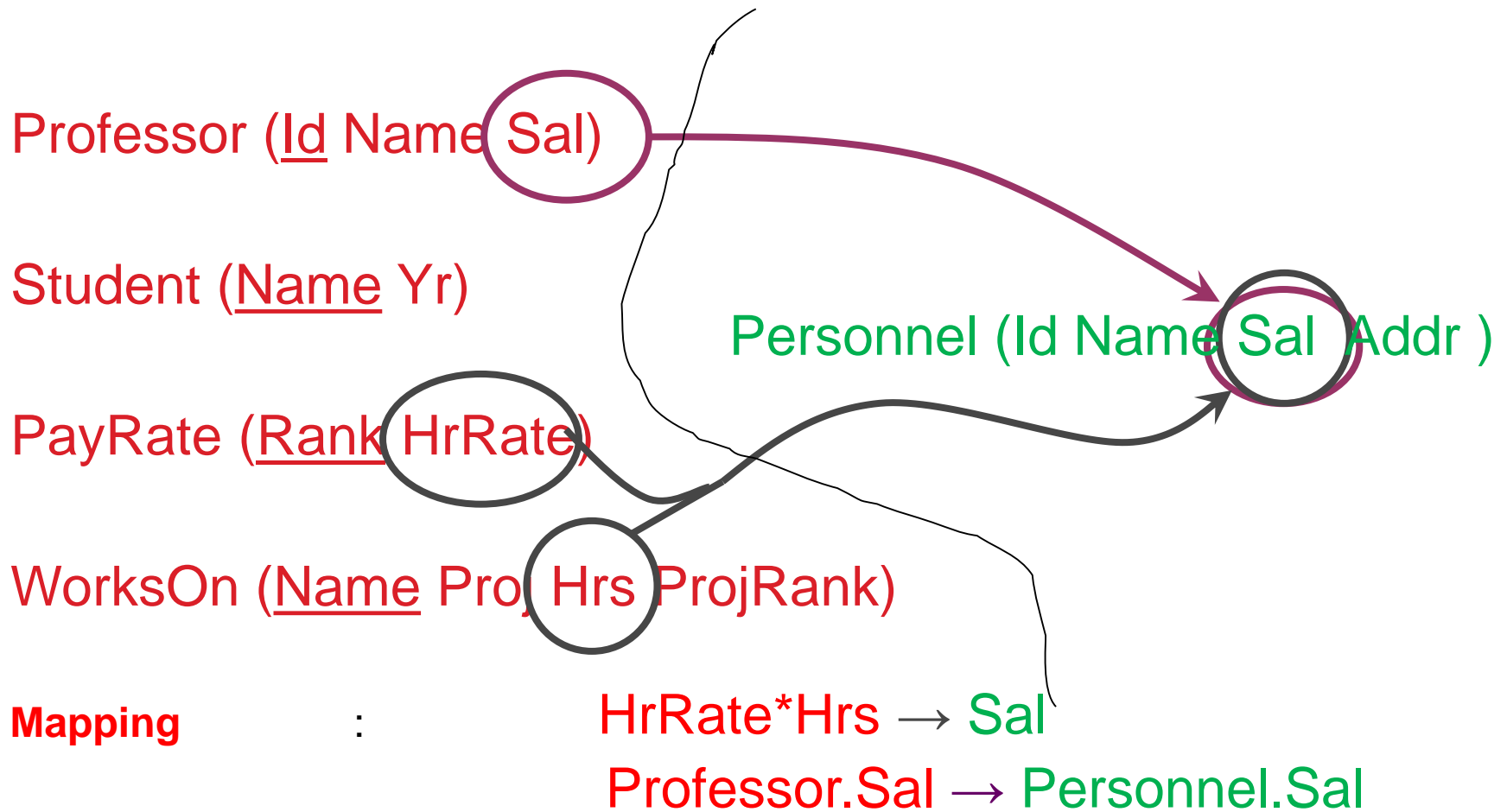
► Exemple (transaction time)

- Les données dans les sources peuvent ne pas être positionnées dans le temps (par exemple, on pourrait savoir qu'il y a eu une vente mais sans connaître la date de cette vente)
- Il est peut être possible de connaître le temps auquel la donnée était insérée dans la source (Tempsinsert)
 - Si la source est une base de données, cela s'appelle le « transaction time » (et peut être connu à travers plusieurs techniques)
- Si ce temps est connu, le temps dans lequel la donnée se positionne pourrait être estimé par
 - Tempsinsert-délaimoyeninsert
 - Où le délaimoyeninsert est l'intervalle de temps nécessaire à transférer vers la source une donnée connue
- D'une manière générale il est intéressant de disposer des ces 2 temps

Echange de données (ED)

- ▶ Dans ce cas, il est supposé de disposer d'un **schéma source** et d'un **schéma cible**
- ▶ Objectif : produire des **données cible** respectant un **schéma cible** à partir de **données source** (exemples : flux d'extraction, flux de chargement, flux SI→SD)
- ▶ Conceptuellement similaire à la **définition de vue dans les bases de données**
- ▶ Nécessite de préciser comment à partir de données sources les données cible sont générées ; cela passe par la spécification de **mappings (exécutables)** ; il est donc usuel d'utiliser des **schémas logiques** pour spécifier l'échange de données

Exemple I



Comment générer la base de données cible à partir d'une base source ? Générer une (ou plusieurs) requête(s) correspondante(s)

Exemple II

On considère le schema source ci-dessous :

Cours (cn, intitulé, credits)

Promo (cn, sem)

Enseignant (id, nom)

Enseigner (id, cn, sem)

Titularité (cn, id)

Clés primaires
surlignées

et le schéma cible ci-dessous :

NSchema (cours, intitulé, nom_ens)

Exemple II

Schéma source :
Cours (cn, intitulé, credits)
Promo (cn, sem)
Enseignant (id, nom)
Enseigner (id, cn, sem)
Titularité (cn, id)

Cours

cn
intitulé
credits

Enseignant

id
nom

Mapping conceptuel (exprimée
comme relations entre colonnes) :

Schéma cible

cours
intitulé
nom-ens

Tables sources

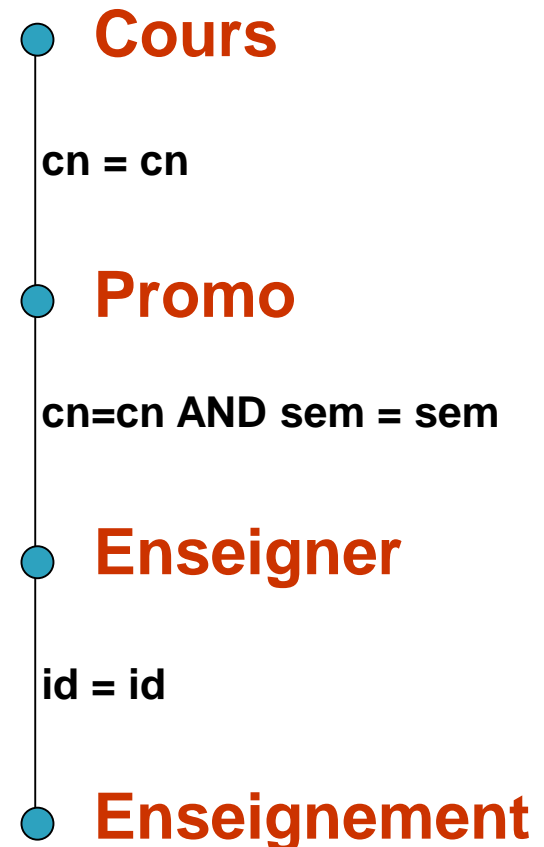
Tables cibles

Génération de la base de données cible I

Mapping exécutable → génération d'une requête

Cours (cn, intitulé, credits)
Promo (cn, sem)
Enseignant (id, nom)
Enseigner (id, cn, sem)
Titularité (cn, id)

Une requête possible (cours enseignés par un enseignant)



Génération de la base de données cible II

Mapping exécutable → génération d'une requête

Cours (cn, intitulé, credits)

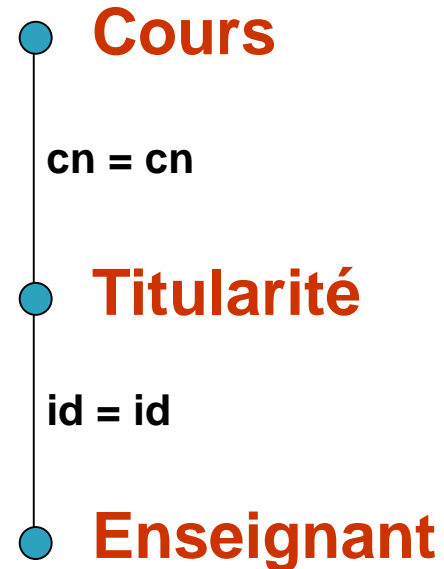
Promo (cn, sem)

Enseignant (id, nom)

Enseigner (id, cn, sem)

Titularité (cn, id)

Une autre requête possible
(cours et les titulaires)



Quelle requête est correcte ? Celle-ci ou la précédente ?

Élimination des erreurs, standardisation de données générées

- ▶ L'échange de données dans le contexte des entrepôts peut être réalisé dans le but **d'améliorer la qualité de données d'origine** ou d'en **standardiser la représentation, suivant les schémas reconstitués des sources**
- ▶ Cela permet de simplifier la successive intégration de données
- ▶ Exemples :
 - Supprimer des espaces, mettre en majuscule, éliminer les accents (standardisation), rajouter ou supprimer des caractères (standardisation/élimination d'erreur)
 - Indiquer une valeur manquante (standardisation), substituer une valeur par une autre (élimination d'erreur)
 - Éliminer les doublons (standardisation)
- ▶ Les mappings permettent d'éviter l'écriture de programmes/requêtes assez longs et articulés

Échanges SQL-Like (Extraction de données pertinentes – Filtrage)

- ▶ Recherche de similarité :
 - Select a,b from T a, S b where **sim**(a,b) > seuil
- ▶ Recherche de données de mauvaise qualité :
 - Select a [ou count(a)] from T where b is null [ou a is null]
 - Select a from T where a not in (...) [ou a > seuil ou a < seuil]
 - Select a, b from T where a > b [ou a + b > seuil, a + n < seuil]
 - Select a from (select a,b from T) group by a having count(a) > 1
 - Select a from T,P p where not **match**(a, p.a)

Échanges SQL-Like (Extraction de données pertinentes – Données extraites)

- ▶ Recodage :
 - `select case when a=v then w when a=v' then w' end from T`
 - `Select F(a) from T`
- ▶ Changement d'unité de mesure-valeur :
 - `select F(a) from T / select F(T.a,S.b) from T,S (join)`
- ▶ Changement de clés-valeur :
 - `Select ID, seq.next from T`
- ▶ Fusion :
 - `select merge(a,b) from T / select merge(a,b) from T, S (join)`
- ▶ Split :
 - `select F1(a), F2(a),... from T`
- ▶ Agrégation :
 - `select OP(a), b from T group by b`
- ▶ Modification :
 - `update T set a=... where`

Échanges non SQL (Extraction de données pertinentes – Données extraites)

```
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('VINET','Vins et alcools Chevalier','59 rue de l'Abbaye','Reims',null,'51100','France');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('HANAR','Hanari Carnes','Rua do Paço, 67','Rio de Janeiro','RJ','05454-876','Brésil');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('VICTE','Victuailles en stock','2, rue du Commerce','Lyon',null,'69004','France');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('SUPRD','Suprêmes délices','Boulevard Tirou, 255','Charleroi',null,'B-6000','Belgique');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('CHOPS','Chop-suey Chinese','Hauptstr. 29','Bern',null,'3012','Suisse');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('HILAA','HILARIÓN-Abastos','Carrera 22 con Ave. Carlos Soublette #8-35','San
Cristóbal','Táchira','5022','Venezuela');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('ERNSH','Ernst Handel','Kirchgasse 6','Graz',null,'8010','Autriche');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('OTTIK','Ottilies Käseladen','Mehrheimerstr. 369','Köln',null,'50739','Allemagne');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('QUEDE','Que Delícia','Rua da Panificadora, 12','Rio de Janeiro','RJ','02389-673','Brésil');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('RATTC','Rattlesnake Canyon Grocery','2817 Milton Dr.','Albuquerque','NM','87110','États-Unis');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('FOLKO','Folk och få HB','Åkergatan 24','Bräcke',null,'S-844 67','Suède');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('FRANK','Frankenversand','Berliner Platz 43','München',null,'80805','Allemagne');
(CODE_CLIENT,SOCIETE,ADRESSE,VILLE,REGION,CODE_POSTAL,PAYS) ('GROSR','GROSELLA-Restaurante','5ª Ave. Los Palos Grandes','Caracas','DF','1081','Venezuela');
```

<<table>>
Ville
ID <<PK>>
Ville
Region
CP
Pays

Utilisation d'une expression régulière en Java pour le CP

"[^\d]*[\d]+([^\d]+)" 2^{ème} numéro ---

"[^\d]*[\d]+([^\d]+[-][\d]+)" numéro suivi de
« - » suivi d'un numéro

Et pour extraire le pays "[^\d]*[\d]+([^\d]+[\d]+(.*))"

Échanges SQL-Like (Extraction de données pertinentes – Données extraites)

- ▶ Les mappings SQL-like fournis comme exemples, peuvent être facilement réalisés par un ETL quelconque
- ▶ Cependant, ces mappings ne prennent pas en considération **2 problématiques majeures** à résoudre lors de la programmation des flux d'extraction
 - Comment réaliser les **extractions successives**
 - Comment prendre en compte des éventuels **changements des clés dans les sources**

Prise en compte des extractions successives

- ▶ Si la source est une base de données (peu importe le modèle de données), plusieurs mécanismes peuvent être mis en œuvre
- ▶ Le plus simple (mais difficile à gérer à cause de l'entrelacement transactionnel) est celui utilisant un trigger vers une table contenant les données modifiées ; par exemple

Create trigger ChangementsT after update or insert or delete on T

For each row

Begin declare r T%ROWTYPE; s varchar(20);

If updating then Insert into ChOfT values (:old.key, :new.a, :new.b,..., 'updated'); s='updated';
r.key=:old.key..... end if; /* il est supposé qu'en cas de changement de clés, la nouvelle donnée correspond à l'ancienne donnée

If inserting then Insert into ChOfT values (:new.key, :new.a, :new.b,..., 'inserted') ; s='inserted'; r=:new;
end if;

If deleting then Insert into ChOfT values (:old.key, :old.a, :old.b, ..., 'deleted'); s='deleted'; r=:old; end if;

Exception when others update ChOfT set a=r.a; b=r.b, ..., status=s where ChOfT.key=r.key end;

- ▶ La table contenant les changements sera donc vidée par un flux ETL une fois pris en compte ces changements

Prise en compte des extractions successives

- ▶ Une solution plus viable pourrait être celle des « log », si disponibles ; par exemple (ORACLE)

```
create table T  
(a number primary key, b varchar2(19))
```

```
create materialized view log on T with rowid, primary  
key, sequence (b) including new values;
```

```
-- select * from MLOG$_T
```

Prise en compte des extractions successives

- ▶ Si la source n'est pas une base de données, la solution ne peut être que spécifique
- ▶ Par exemple, si un fichier texte est utilisé,
 - il faudra probablement comparer l'ancien fichier au nouveau fichier pour repérer les changements ou
 - Limiter la comparaison rajoutant dans la source une information sur la prise en compte par le flux ETL des données contenues
- ▶ Évidemment, il est toujours possible de vider entièrement la zone de staging et refaire une extraction complète mais cette solution pourrait ne pas être possible et pourrait ne pas être scalable

Échange de données dans l'approche « supply driven »

Identification sources	Conception SI	Conception SD	Conception Flux
Analyse (statique) de qualité : choix de sources	Transf Schéma (C→C)	Transf Schéma (C→L;C→C;C→L)	Conception de flux d'extraction : Échange de données visant l'amélioration de la qualité (mappings conceptuels)
Transf Schéma (L→C;C→L)	Intégration schéma (conceptuel)+Transf Schéma (C→L)		Intégration de données (conceptuel)

Conception

Réimplantation sources	Implantation SI	Implantation SD	Programmation flux
Transf Schéma (L→P sources) + déploiement	Transf Schéma : L→ P schéma intégré) + déploiement	Transf Schéma : L-> P, L→ L schéma dimensionnel)+ déploiement	Flux de chargement : Échange de données (chargements successifs)+ déploiement (parallélisation)
			Flux d'intégration + déploiement (parallélisation)
			Flux d'extraction : Échange de données (mappings exécutables et extractions successives)+ déploiement (parallélisation)

Mise en œuvre

Points clé :

- Les échanges en rouge sont complexes ; plus ces échanges sont soignés, plus l'intégration de données sera simple ;
- Les échanges en vert sont généralement plus simples car bien cadrés et sous l'hypothèse de disposer d'outils adaptés

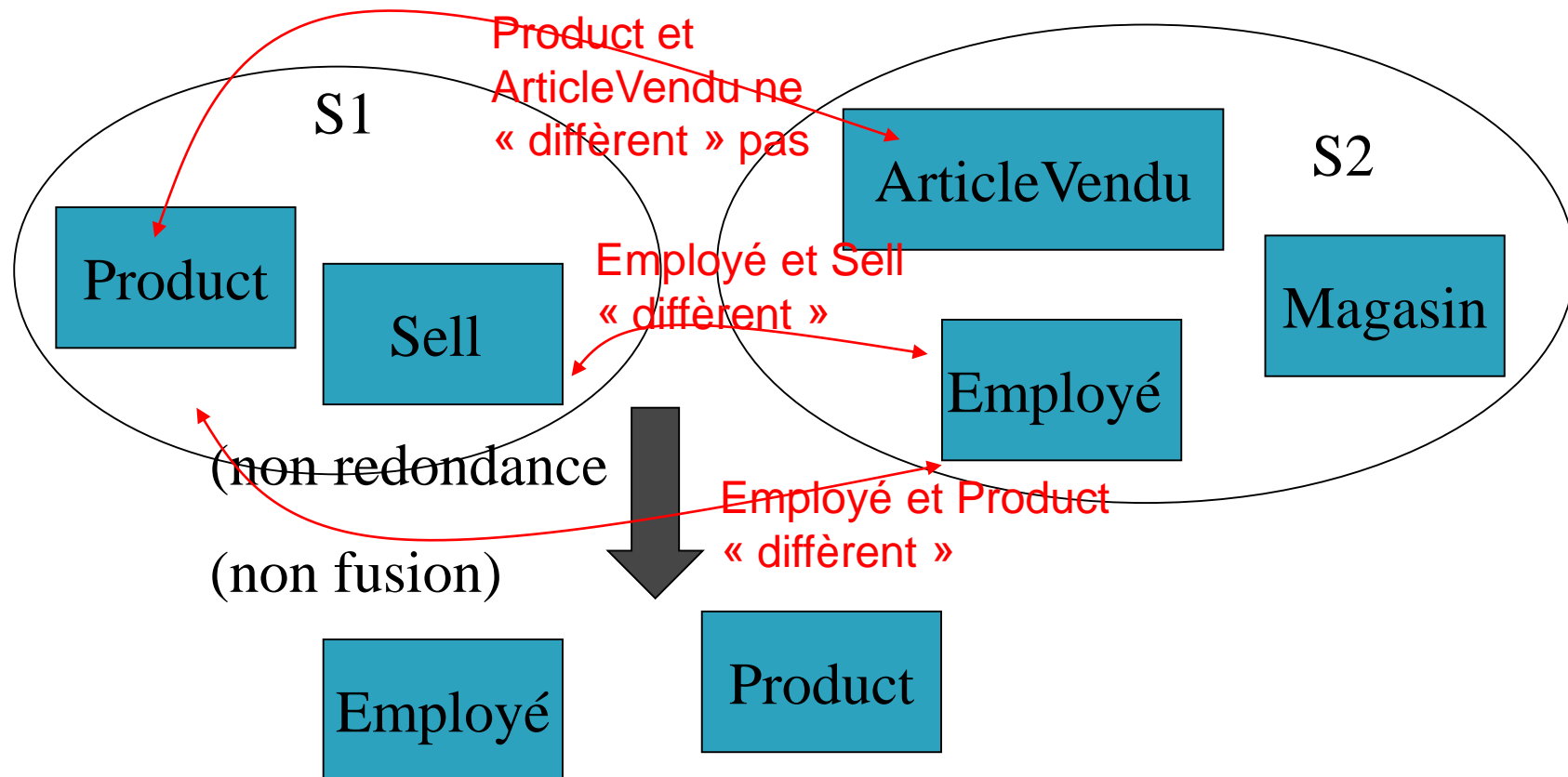
Intégration de schémas

- ▶ Objectif : Étant donné un ensemble de schémas en entrée, produire **un seul schéma dit intégré** tel que
 - **toute (et seule) l'information associée et associable aux schémas en entrée** (capacité de décrire toutes les données, les contraintes etc.) est représentable dans ce schéma
 - **Limite ou exclu la « redondance » de données**
 - **Limite ou exclu la « fusion erronée » de données**
 - **la « granularité » des schémas en entrée est maintenue (pas d'agrégats de données)**
- ▶ Cependant, lors d'une intégration de schémas :
 - On peut exceptionnellement exclure d'une manière explicite certaines données (cela devrait être fait en priorité lors de l'identification de sources et des transformations de leurs schémas)
 - Par conséquent, on peut rajouter des nouvelles contraintes ou exclure certaines données (données inutiles ou inutilisables)

Questions typiques lors d'une IS

Comment décrire différences et similitudes entre schémas (**correspondances inter-schéma**) ; que signifie que 2 schémas sont différents ou pas ?

Comment bien décrire les différents schémas pour les préparer à l'intégration ?



Signification d'une correspondance : la relation

Employé
salaire

Projet
ProjetID

S1

Employé			
ID	Salaire		S1
1	100		
2	200		
3	500		

Relation			Projet
ID1	ID2		ID Budget
1	2		1 10000
2	2		2 20304
3	1		3 564451

Passé

- ▶ Employés in S1 = Employés in S2 ?
- ▶ Employés in S1 contient Employés in S2 ?
- ▶ Qui seraient les mêmes employés ?
- ▶ → est ce que ces mêmes employés auraient le même salaires ? OU
- ▶ → est ce que les mêmes employés seraient ceux ayant le même salaire ?
- ▶ → est ce que les mêmes employés seraient ceux travaillant sur les mêmes projets ? Dans ce cas, il faut aussi pouvoir préciser quels seraient les mêmes projets

Relation			Employé
ID1	ID2		ID Salaire
1	2		1 150
2	1		2 180
3	2		3 590

Team			
ID1	ProjetID		S2
1	5		
2	1		
3	2		

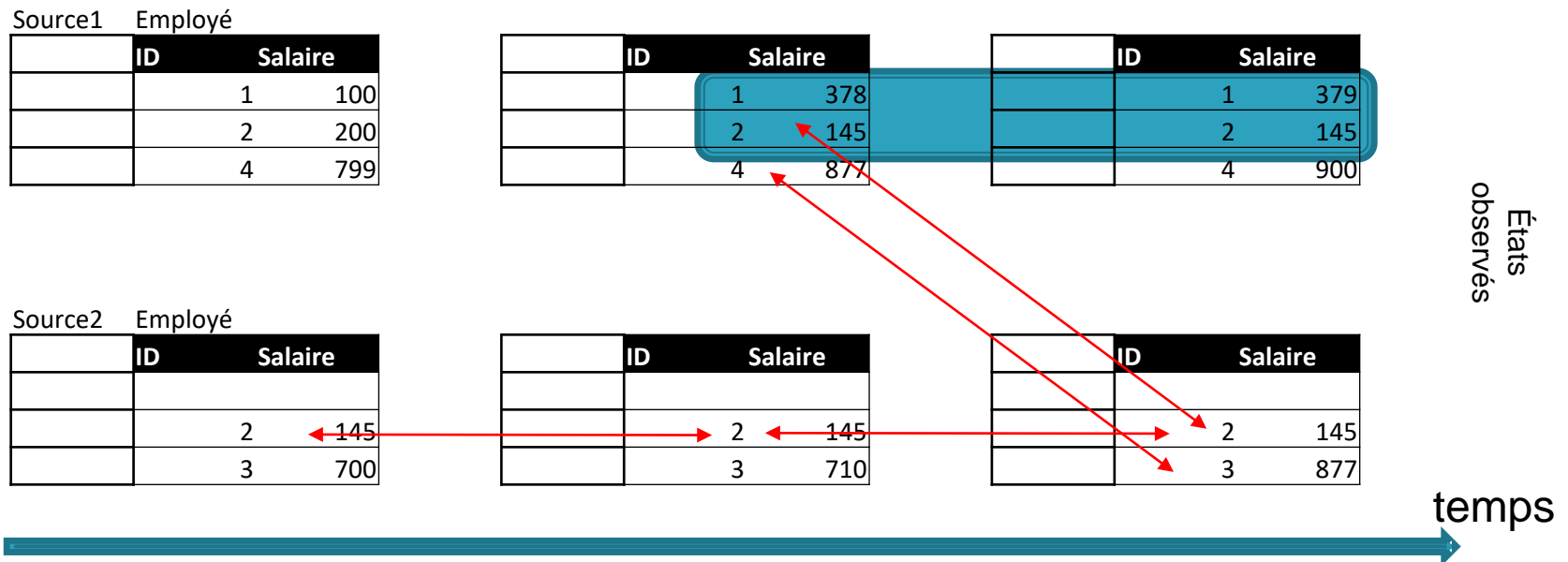
Présent

Employé
salaire

Team
ProjetID

S2

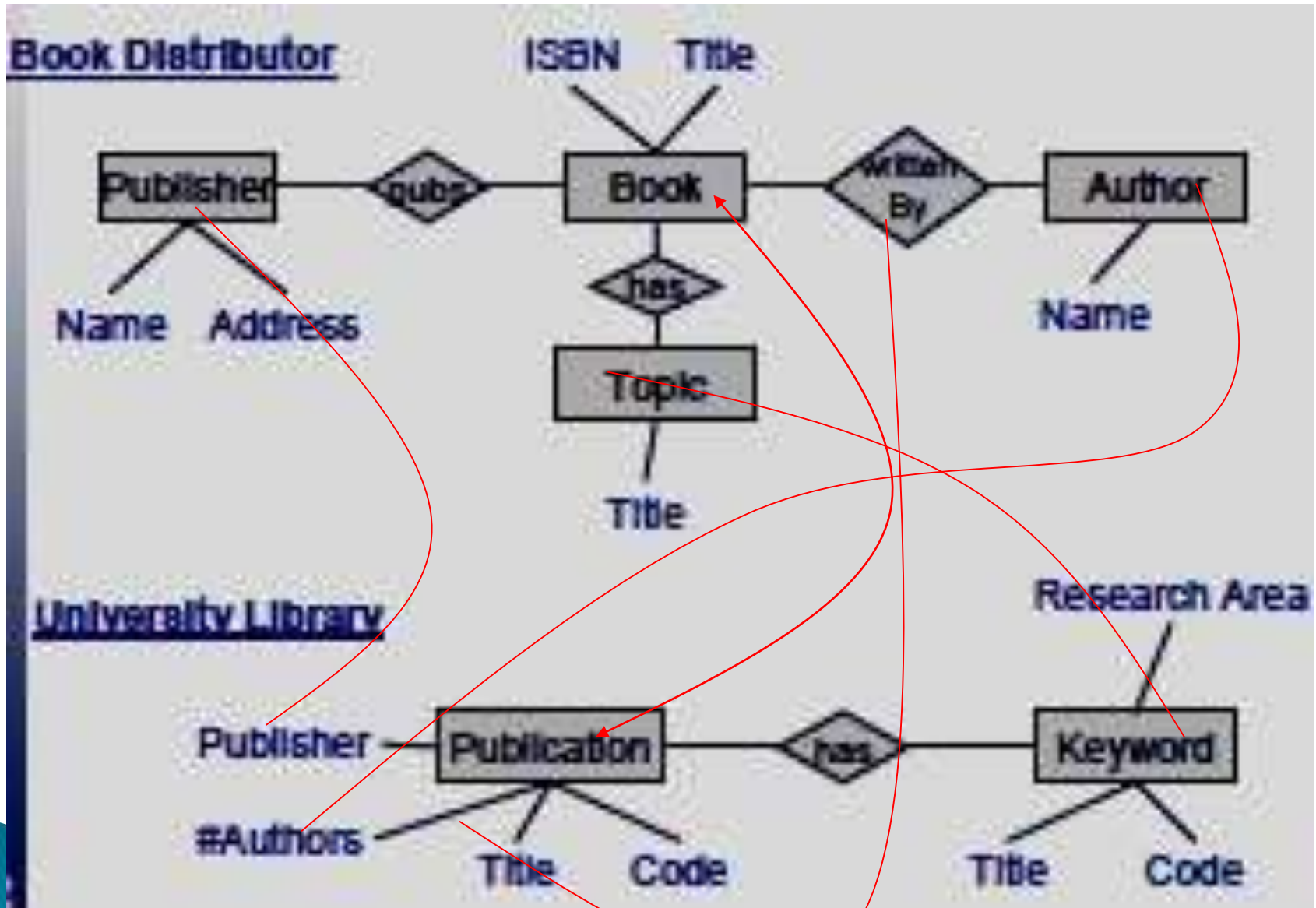
Signification d'une correspondance : la relation et les états observés



Flèche = données correspondantes à **une même réalité ou à un même état de cette réalité**

Il n'est pas possible de spécifier une fonction de correspondance (injective) liant ces données et prenant en compte que ces données

Un exemple plus complexe



Périmètre d'une correspondance

- ▶ D'une manière erronée, l'on pourrait penser que les correspondances s'établissent entre structures proches ou égales
- ▶ Cela n'est pas vrai, comme le montre l'exemple précédent
 - Par exemple un attribut possédant une valeur peut en effet représenter un objet dont la valeur de cet attribut n'est qu'un moyen d'identification (par exemple attribut « Auteur », transparent précédent)
- ▶ Les correspondances sont donc définies entre **objets** même si ces objets ne sont pas représentés explicitement dans les schémas
 - Il n'est pas très difficile de formaliser cet aspect en transformant les schémas à intégrer pour représenter explicitement ces objets (**objectification**)

Formalisation d'une correspondance

S1.Employé

ID	NOM	SAL

ID	NOM	SAL

S2.Employé

C :

S1.Employé \equiv S2.Employé

sous condition d'identification

S1.Employé.ID=S2.Employé.ID

alors

S1.Employé.Sal=S2.Employé.Sal

$\forall x \quad \forall y \quad \forall z \quad (f2(x)=y \text{ and } S2.employé(y) \text{ and } f1(x)=z) \rightarrow S2.employé(z))$ } La relation

$\forall x \exists y \exists z \quad ((f1(x)=y \text{ and } S1.employé(y) \text{ and } f2(x)=z \text{ and } S2.employé(z) \text{ and } y.ID=z.ID) \rightarrow x.sal=y.sal)$ } Généralisation de la notion de dépendance fonctionnelle

La correspondance est **satisfiable ssi les formules sont satisfiables** (ils existent Ω – ensemble d'objets –, $f1, f2: \Omega \rightarrow \Omega$ – fonctions – tels que $\Omega, f1, f2 \models C$)

En pratique...

S1.Employé \equiv S2.Employé
sous condition d'identification
S1.Employé.ID=S2.Employé.ID
Alors
S1.Employé.Sal=S2.Employé.Sal

- ▶ Si un même employé en commun existe, alors celui-ci devra avoir le même ID une fois projeté dans les 2 sources et, par conséquent, il devra avoir le même salaire
 - La condition d'identification et ses conséquences garantissent l'objectif de limiter ou exclure la redondance
- ▶ Parmi ces employés en commun, il y en a qui apparaissent dans la source S1 mais qui n'apparaissent pas dans la source S2

En pratique...

$S1.\text{Employé} \sqsupseteq S2.\text{Employé}$
sous condition d'identification
 $S1.\text{Employé.ID} = S2.\text{Employé.ID}$
Alors
 $S1.\text{Employé.Sal} = S2.\text{Employé.Sal}$

- ▶ 2 employés trouvés dans les 2 sources, **pourraient être la même personne** si leurs ID (valeur) observés dans les 2 sources est le même
 - Mais la condition d'identification ne garantit pas de trouver les mêmes employés
- ▶ Si l'on confirme que 2 employés ayant le même ID sont la même personne alors leur **salaire devra être aussi le même** (indépendamment de ce que l'on peut observer à un même instant dans les 2 sources) ; autrement dit, un même employé ne peut pas **disposer de 2 salaires distincts**
- ▶ Et sous ces conditions, dans S2 **on trouvera (globalement) moins** d'employés que dans S1

Forme d'une correspondance inter-schéma (cadre syntaxique)

$S1.C1^{(type1)}$ *set_relationship* $S2.C2^{(type2)}$ (les artefacts correspondants, étant $S1$, $S2$ les 2 schémas)

- [WC Identifiers (comment identifier les objets correspondants ou devant correspondre entre $S1.C1$ et $S2.C2$) :
 - $C1.id-predicate = C2.id-predicate^{(1)}]_{disjoint}^{(2)}$
- [WC Property (comment des propriétés possédées par ces objets correspondants sont liées) :
 - $C1.property \ set_relationship \ C2.property, \dots^{(1)}]_{disjoint}^{(2)}$

Où *set_relationship* vaut : EQUAL, CONTAIN, INTERSECT, DISJOINT

(1) id-predicate/property est une expression SQL-like, logiques etc. exprimée sur chaque schéma

(2) Non applicable si $C1$ et $C2$ sont disjoints

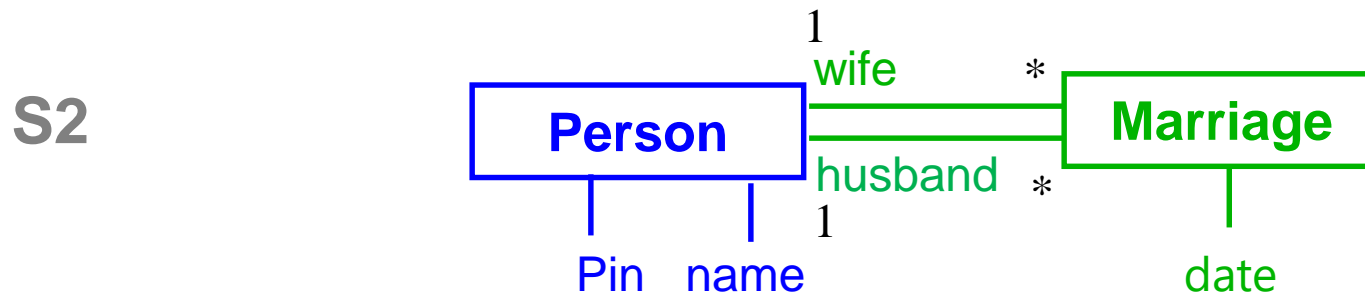
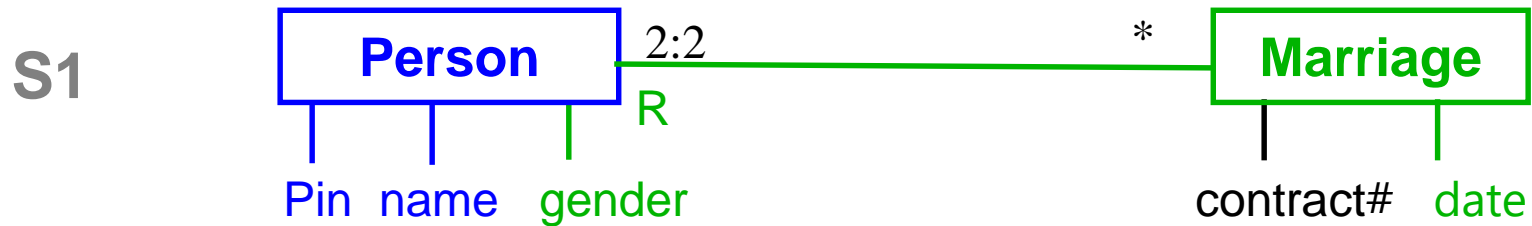
Utilité d'une correspondance interschéma

- ▶ Formulation rigoureuse **d'une hypothèse** entre 2 schémas (éventuellement dans un même langage, conceptuels ou logiques) portant sur les **équivalences, similitudes, différences** entre 2 schémas sur la base de données possibles dans le but de la **vérification de non contradiction (logique)**
 - **Difficulté théorique (formalisation en logique, satisfiabilité)**
- ▶ Détermination des **structures conflictuelles** entre ces schémas et pour construire un schéma intégré
- ▶ Point de départ pour spécifier les **flux d'intégration de données**

Cas de correspondance interschéma

- ▶ Chaque élément d'un schéma peut être perçu comme **une classe d'objets (objectification)**
- ▶ La **syntaxe des correspondances** s'inspire à la navigation dans un **schéma objet-relationnel** (notation à point, complétée par la sélection)
 - Mais il n'y a pas besoin d'utiliser un opérateur fictif (tel que « table »), les relations sont navigables dans les 2 sens et les attributs objectifiés sont navigables à l'inverse
 - pour une syntaxe précise : https://moodle.univ-ubs.fr/pluginfile.php/361469/mod_resource/content/3/SchemaCorrSpec.pdf ; une formalisation en logique est aussi possible, utilisant par exemple une logique de description)
- ▶ Cas présentés (en langage similaire à UML) :
 - Classe/Classe
 - Classe/Association
 - Classe/Attribut
 - Association/Association

Exemple I



- S1.Person Equal S2.Person WCI:
 Person.Pin.value=Person.Pin.value WCP:
 Person.name.value=Person.name.value
- S1.Marriage Equal S2.Marriage,
 WCI: S1.Marriage.R. σ [gender="F"] = S2.Marriage.wife,
 S1.Marriage.R. σ [gender="M"] = S2.Marriage.husband,
 WCP: Marriage.date=Marriage.date
 Note : le WCI identifie des objets
 correspondants via des relations/associations,
 utilisant une connaissance externe à S2

Exemple I (bis)

► Changement de la relation

- $S1.\text{Marriage}$ **CONTAIN** $S2.\text{Marriage}$

- WCI:

$S1.\text{Marriage}.R.\sigma [\text{gender}=\text{"F"}] = S2.\text{Marriage}.wife,$
 $S1.\text{Marriage}.R.\sigma [\text{gender}=\text{"M"}] =$
 $S2.\text{Marriage}.husband,$

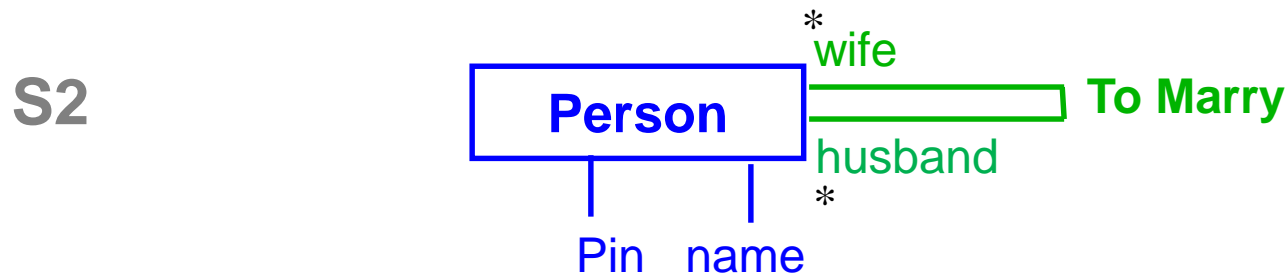
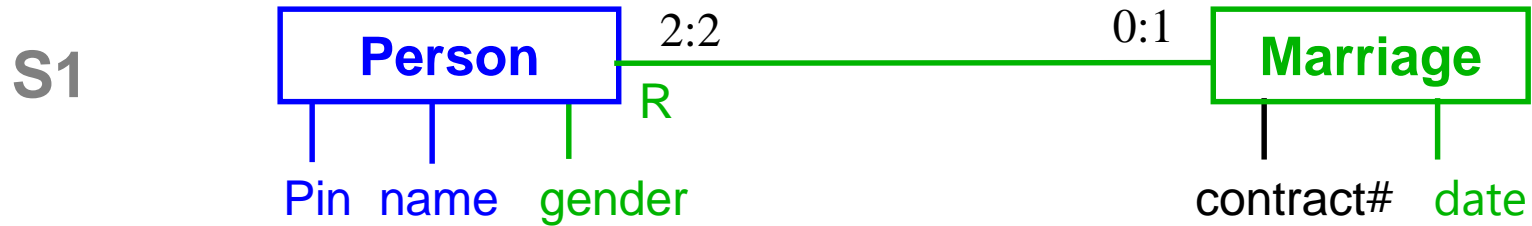
- WCP: $\text{Marriage.date} = \text{Marriage.date}$

► Transfert WCP/WCI

- $S1.\text{Person}$ Equal $S2.\text{Person},$

- WCI: $\text{Person.Pin.value} = \text{Person.Pin.value},$
 $\text{Person.name.value} = \text{Person.name.value}$

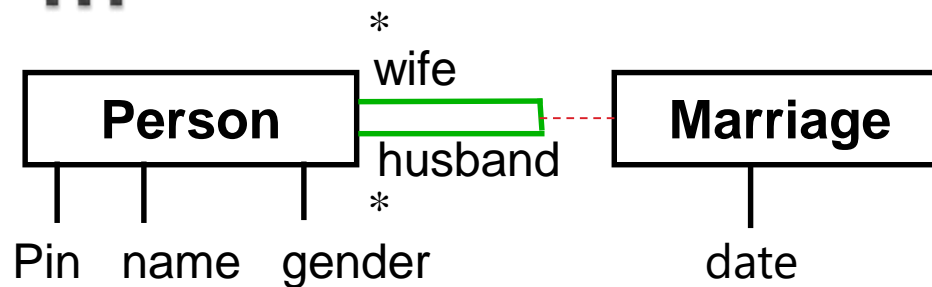
Exemple II



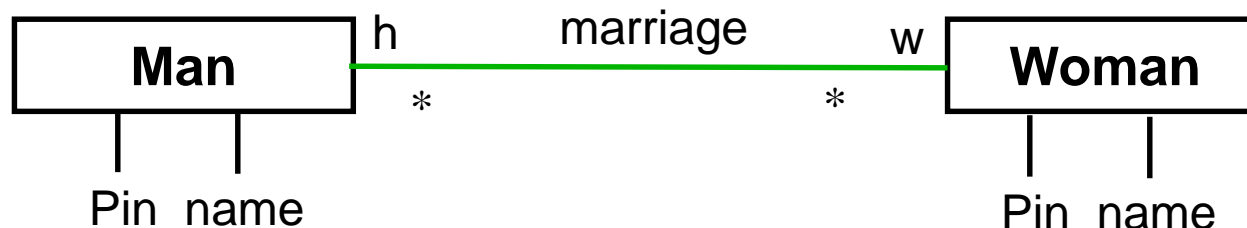
- S1.Person Equal S2.Person, WCI: Pin, WCP: name
- S1.Marriage Equal S2.ToMarry,
 - WCI: S1.Marriage.R.σ [gender="F"] = S2.ToMarry.wife,
S1.Marriage.R.σ [gender="M"] = S2.ToMarry.husband,
 - WCP ...

Example III

S1



S2



S1.Person Equal S2.(Man U Woman) ←

WCI: Pin WCP: name

Artefact constructible
(voir DL)

S1. Person. σ [gender="M"] Equal S2.Man

WCI: Pin WCP: name

S1. Person. σ [gender="F"] Equal S2.Woman

WCI: Pin, WCP: name

S1. Person Contain S2.Woman,

WCI: Pin WCP: name

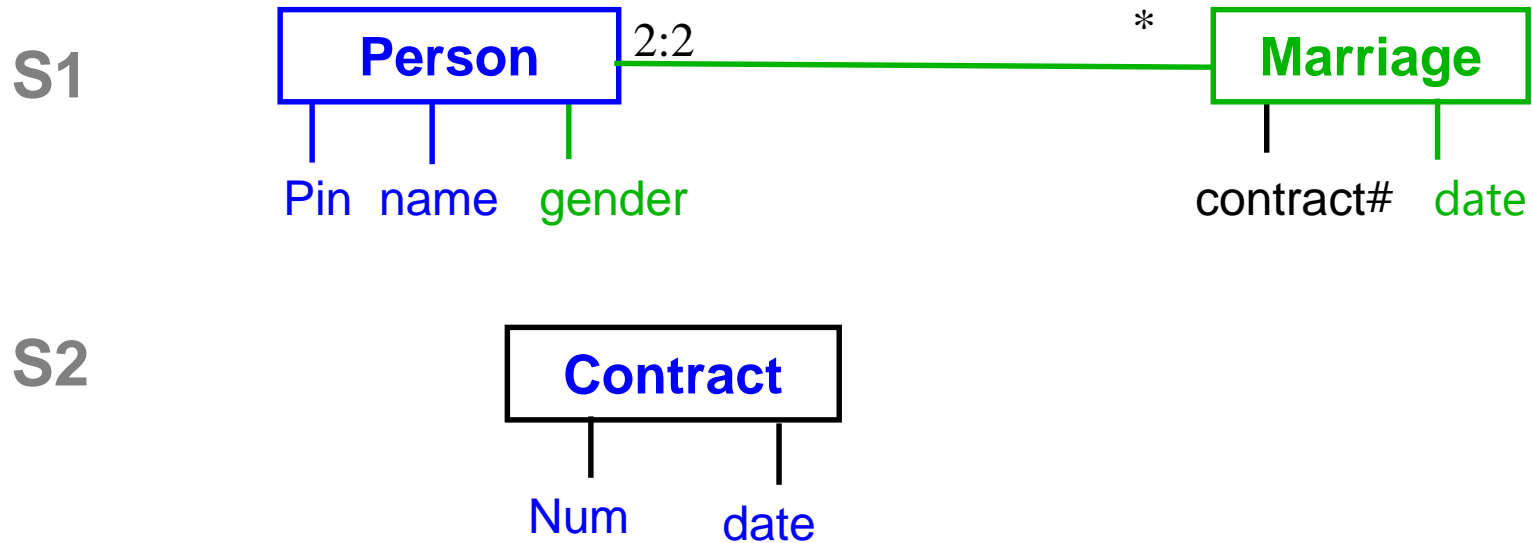
S1. Person Contain S2.Man,

WCI: Pin WCP: name

Correspondances et artefacts constructibles

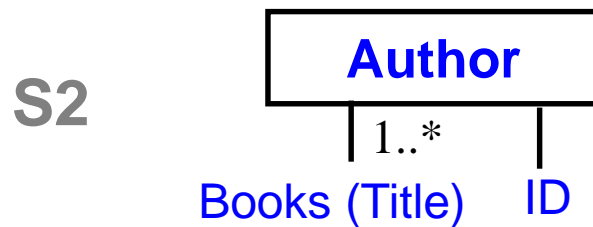
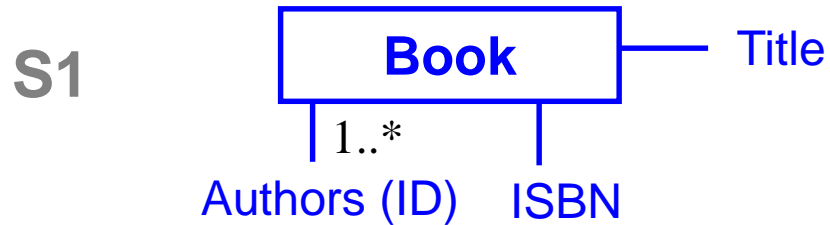
- ▶ L'exemple précédent montre l'intérêt d'introduire des correspondances entre relations
- ▶ Cela pourrait être étendu aux navigations (ou relations composées) pour créer des « link correspondance » plus puissants
- ▶ Comme déjà remarqué, il est possible d'étendre les correspondances à tous les **artefacts constructibles** à partir d'autres artefacts

Example IV



- S2.Contract CONTAIN S1.Marriage
 - WCI: S1.Marriage.contract#.value=S2.Contract.Num.value
 - WCP: S1.Marriage.date=S2.Contract.date
- S2.Contract CONTAIN S1.contract#,
 - WCI: S1.contract#.value=S2.Contract.Num.value
 - WCP: S1.contract#.Marriage.date=S2.Contract.date

Exemple V



- S2.Books EQUAL S1.Book
 - WCI: $S2.Books^{-1}.Author.ID = S1.Book.Authors.value$, $S1.Book.Title = S2.Books.value$
 - WCP: $S2.Books^{-1}.Author$ CONTAIN $S1.Book.Authors$
- S2.Author CONTAIN S1.Authors
 - WCI: $S1.Authors.value = S2.Author.ID$

Example VI



S1.R1 equal S2.R2

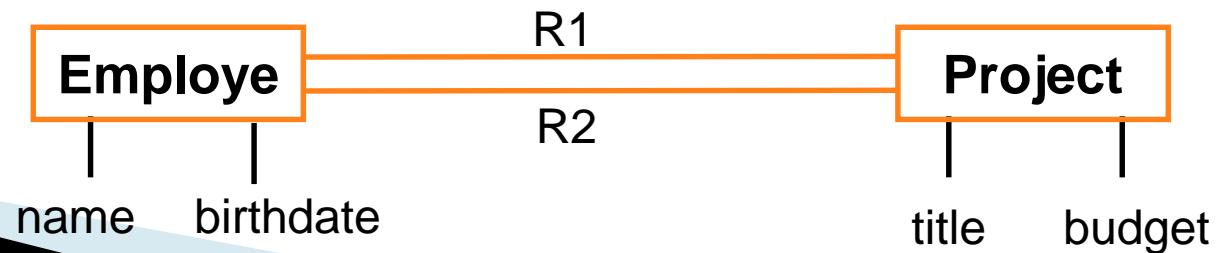
WCI

S1.R1.Employee=R2.S2.Employee

S1.R1.Projet=S2.R2.Project



S1.R1 disjoint S2.R2



Étapes (bases) de l'intégration de schémas

- 1) Recherche des correspondances
- 2) Identification des structures conflictuelles
- 3) Résolution de conflits (réconciliation)
- 4) Intégration de schémas non conflictuels

Recherche de correspondances

► Manuelle

- Exemples de conseils : https://moodle.univ-ubs.fr/pluginfile.php/361469/mod_resource/content/3/SchemaCorrSpec.pdf

► Outillée (semi-manuelle)

- Annotation manuelle (par dictionnaire/ontologie) → SQL Developer permet cela
- Recherche semi-automatique

3 techniques de base pour rechercher des correspondances

- ▶ On intègre différentes structures ayant vocation à enregistrer des informations faisant référence à une même réalité
- ▶ Les données ne sont pas considérées pour avoir une évidence de cette vocation
 - Des synonymes des étiquettes des classes ou entités sont cherchés dans des dictionnaires (mais *Personnel* n'est pas forcément considéré un synonyme de *Employé*)
 - Même choses pour les attributs (mais *Person(code, cost)*, *Product(code, cost)* n'ont pas des liaisons)
- ▶ Les données disponibles sont considérées
 - Les données observées peuvent fournir une évidence de la correspondance entre *Product* et *ArticleVendu* ; en pratique, on construit la fonction de correspondance entre états observés sous l'hypothèse que les données identifiants ne se modifient pas dans le temps
 - Mais les données stockées dans des bases différentes, même utilisant des mêmes valeurs, ne font pas forcément référence au même état de la réalité et/ou à la même réalité (dans *Product* on trouve les produits qui ne sont plus en ventes, dans *ArticleVendu* on ne trouve que les produits vendus mais les valeurs utilisés peuvent être les mêmes)
- ▶ Des mécanismes hybrides sont possibles

3 techniques de base pour rechercher des correspondances

realestate.com

listed-price	contact-name	contact-phone	office	comments
\$250K	James Smith	(305) 729 0831	(305) 616 1822	Fantastic house
\$320K	Mike Doan	(617) 253 1429	(617) 112 2315	Great location

homes.com

sold-at	contact-agent	extra-info
\$350K	(206) 634 9435	Beautiful yard
\$230K	(617) 335 4243	Close to Seattle

- If we use only labels
 - contact-agent matches either contact-name or contact-phone
- If we use only data values
 - contact-agent matches either contact-phone or office
- If we use both labels and data values
 - contact-agent matches contact-phone

Recherche semi-automatique des correspondances : techniques

- ▶ La recherche peut être abordée d'une manière **semi-automatique** par plusieurs techniques ; des exemples sont listés ci-dessous

String-based (e.g., **COMA**, **SF**, **S-Match**, **OLA**, **Anchor-Prompt**)

Language-based (e.g., **COMA**, **Cupid**, **S-Match**, **OLA**)

Constraint-based (e.g., **OLA**, **COMA**)

Linguistic resources (e.g., **Artemis**, **S-Match**, **OLA**, **Cupid**, **COMA**)

Alignment reuse (e.g., **COMA**, **COMA++**, **OLA**)

Taxonomy-based (**Anchor-Prompt**, **NOM**, **QOM**)

Graph-based (e.g., **Cupid**, **COMA**, **SF**, **OLA**)

Model-based (e.g., **CtxMatch**, **S-Match**)

Voir [Schema matching survey](#) (Shvaiko & Euzenat. A Survey of Schema-Based Matching Approaches. Journal on Data Semantics IV
Lecture Notes in Computer Science 2005, Volume 3730/2005)
Voir aussi [Schema matching ten years later](#)

Intérêt de l'analyse/exploration de données dans la recherche de correspondance

- ▶ Formulation de la correspondance (Hypothèse)
- ▶ Construction d'une requête d'analyse type
Select WCP
From S1.E1, S2.E2 (join on WCI)
- ▶ Exécution dans l'environnement ETL
- ▶ Avons-nous une règle pour « fusionner » ce qui est égalisé dans le WCP ?
 - Oui, alors on valide le WCP
 - Non, alors on supprime le WCP (mais quelqu'un d'autre pourrait disposer de la règle)

Structures conflictuelles (conflits)

- ▶ Structurel
- ▶ Classification
- ▶ Descriptif

Conflit structurel

- ▶ Les artefacts correspondant par EQUAL, CONTAIN ou INTERSECT, sont associés à des types différents dans les schémas respectifs ; cela introduit des modélisations différentes mais liées
- ▶ Dans l'exemple "marriage I" :
 - S1.Marriage EQUAL S2.Marriage
 - S1 : Marriage est une **entité (classe)**
 - S2 : Marriage est une **relation (association)**

Conflit de classification

- ▶ Les artefacts correspondants représentent des ensembles d'objets, étant ces ensembles à la fois différents mais se superposant (CONTAIN, INTERSECT) ; cela introduit une ambiguïté sur la classification d'un même objet
- ▶ Dans l'exemple "marriage I(bis)":
 - S1.Marriage CONTAIN S2.Marriageimplicitement introduit une double classification pour un mariage mais il n'est pas du tout connu si un mariage doit être dans l'un ou dans l'autre

Conflit descriptif

- ▶ **Les artefacts correspondants, tout cas,** portent des caractéristiques, identifiants et propriétés différents ou des propriétés correspondantes décrites différemment pour ces artefacts
 - ***Conflit de nom: les artefacts correspondants sont étiquetés par de noms contredisant la correspondance***
 - Noms différents, Synonymie (EQUAL)
 - Mêmes noms, Homonymie (DISJOINT, INTERSECT, CONTAIN)
 - ***Conflit de composition: les artefacts correspondants (EQUAL, INTERSECT, CONTAIN) sont équipés de propriétés différentes;*** par exemple :
 - S1.Employee (E# , name , address , job)
 - S2.Employee (E# , position , salary , department)
 - S1.Employee INTERSECT S2.Employee

Conflit descriptif

- **Conflit de cardinalité** : propriétés correspondantes sont associées à cardinalités différentes
- **Conflit de domaine (type)**: propriétés correspondantes sont associées à valeurs/échelles différentes, par exemple :
 - salaire : \$, Euro ...
 - grade ou note étudiant: [0 : 20] , [1 : 5] , [A: D] ...
 - coordonnées : échelle, système de référence, ...
- **Conflit de contrainte** : les artefacts ou les propriétés correspondantes introduisent des contraintes différentes, par exemple
 - La surface minimale d'un appartement (chambre) au Japon ou US ou France n'est pas la même
 - L'âge légal du mariage

Intégration de schémas non conflictuels

- ▶ Une fois complété les précédentes étapes, d'abord il faut donner solution à tous les conflits identifiables avec des règles très simples, par exemple :
 - S1.C1 EQUAL S2.C2 donne lieu à un conflit des noms, alors il faut renommer au moins un artefact dans un des schémas
 - S1.C1 EQUAL S2.C2 et autres donnent lieu à un conflit de cardinalité, alors dans un des schémas il faut utiliser la cardinalité moins contraignante
 - Plus en général, il faut garder les plus de détails possibles
- ▶ Ces règles permettent une solution systématique à tous les conflits et les schémas deviennent non conflictuels ; appliquant les règles tous les conflits disparaissent permettant d'intégrer les schémas par des règles d'intégration des schémas non conflictuels

Règles pour la solution de conflits

► Structurel

- Il faut décider un seul type pour tous les schémas ; ce type remplacera les types d'origine dans tous les schémas ; il est toujours possible de retenir le “type le plus expressif” par exemple classe vs attribut et association, association vs attribut etc.

► Classification

- Le conflit ne sera pas entièrement résolu sauf cas spéciaux où l'on sera capable de reconnaître la classification faite et à suivre

► Descriptif

◦ Nom

- Il faut renommer les artefact en conflit (et faire les remplacements dans tous les schémas)

◦ Cardinalité

- Il faut choisir la cardinalité la moins contraignante (et l'utiliser en substitution dans tous les schémas)

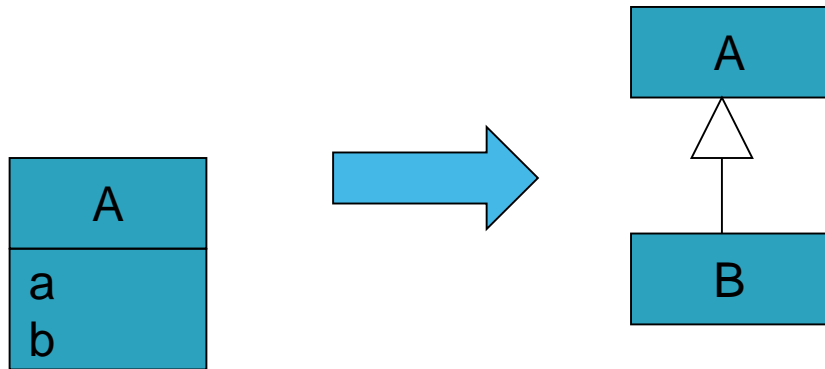
◦ Domaine

- Il faut choisir un domaine (et l'utiliser en substitution dans tous les schémas) ou renommer pour distinguer les éléments conflictuels

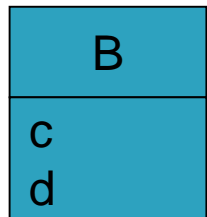
◦ Contrainte

- Il faut rajouter toutes les contraintes dans tous les schémas et les combiner si possible ou pour ne garder que les contraintes les moins sélectifs

Règles (de base) pour intégrer des schémas non conflictuels

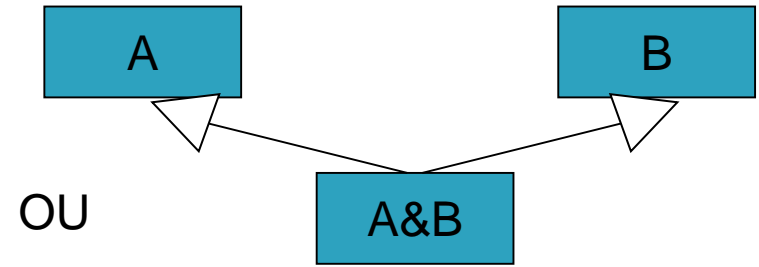


S1.A CONTAIN S2.B



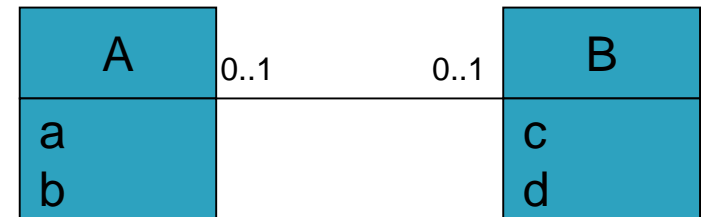
S1.A INTERSECT S2.B

Nota bene : Les diagrammes UML utilisés ne contiennent pas d'opérations (car ils ne représentent qu'un descriptif de données) ET l'identification des objets n'est pas précisée car non nécessaire en UML (par ailleurs, on peut supposer que seul de clés de substitution sont utilisées)

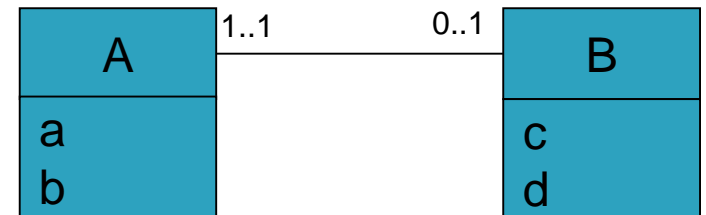


OU

S1.A INTERSECT S2.B

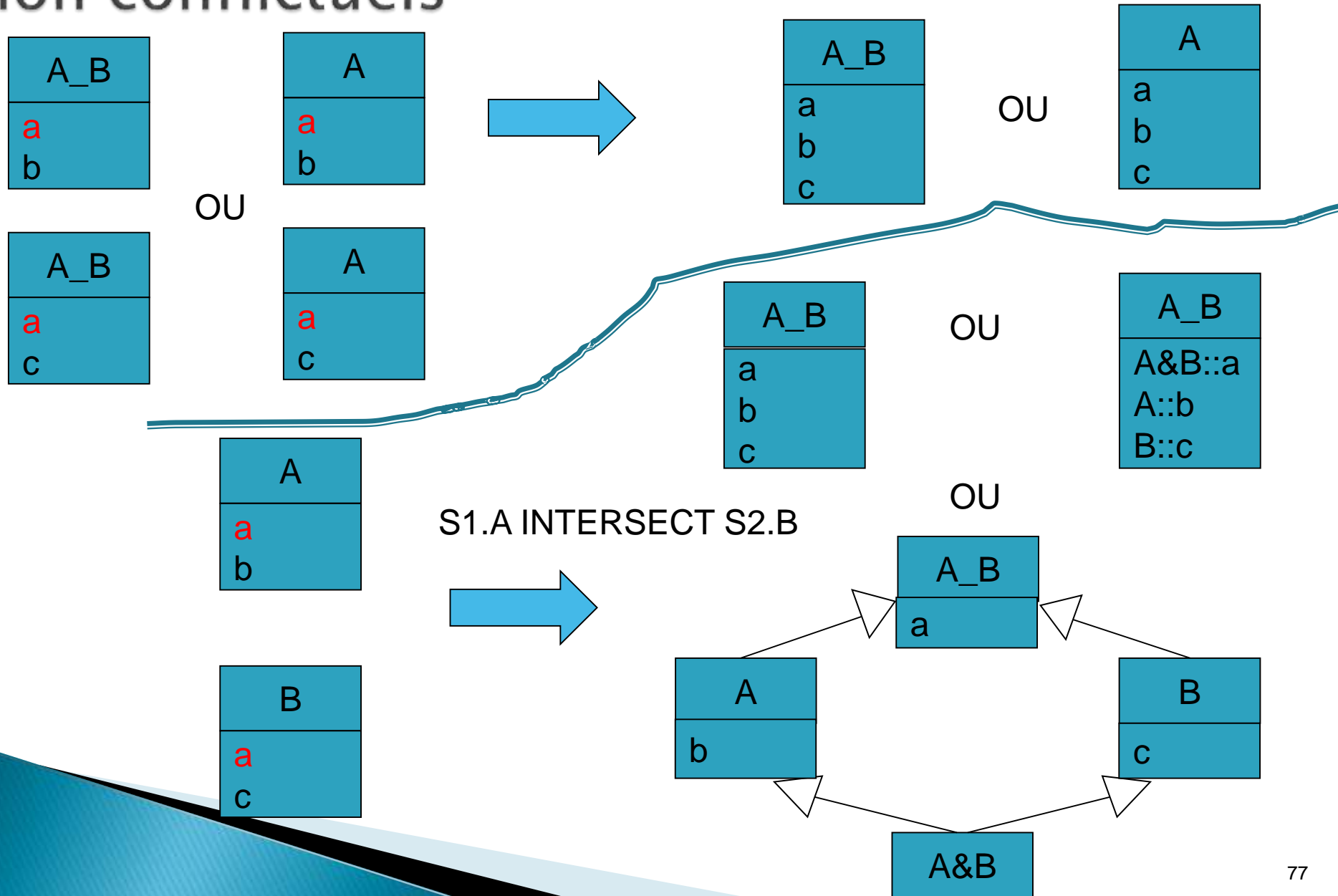


OU

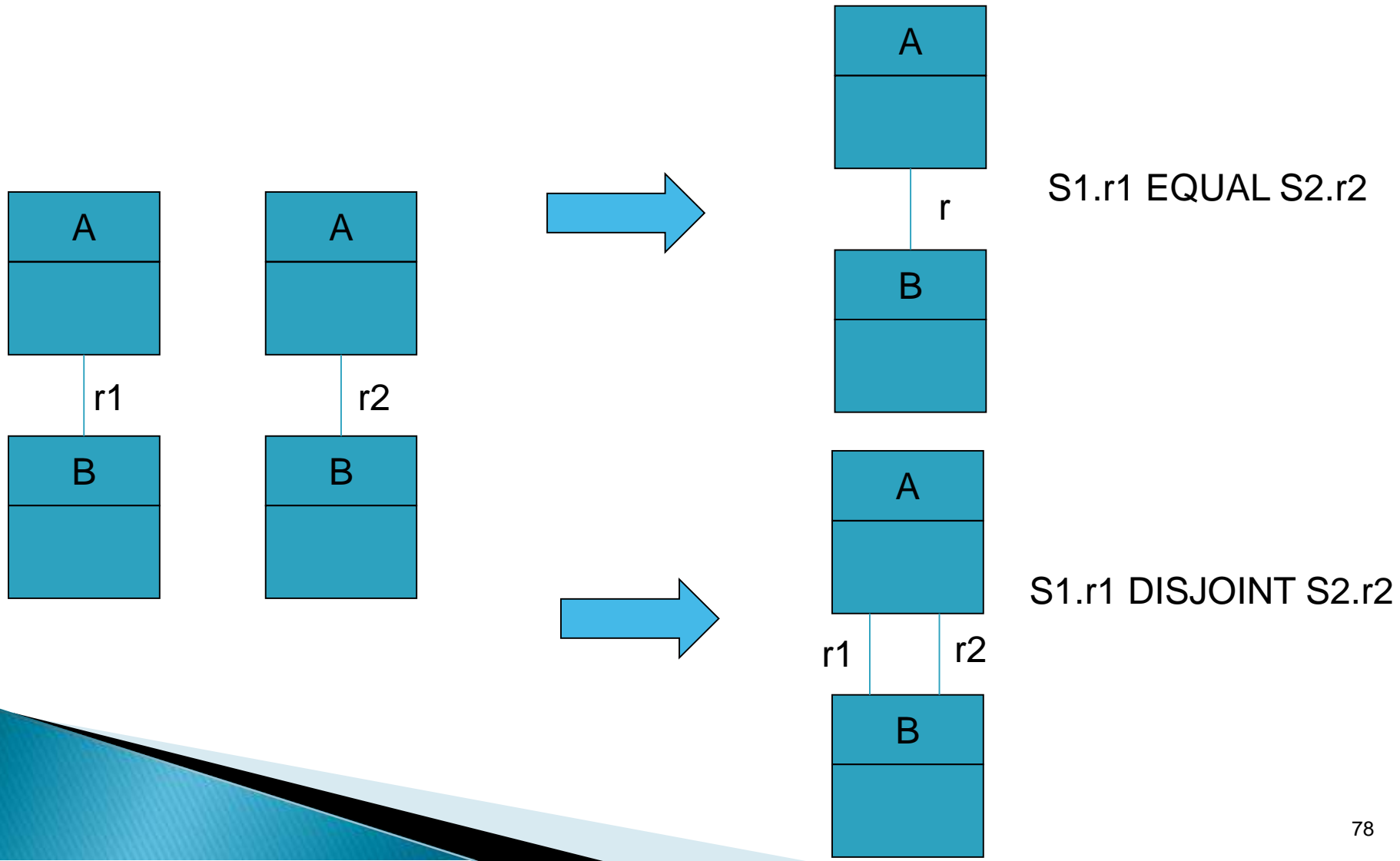


S1.A CONTAIN S2.B

Règles (de base) pour intégrer des schémas non conflictuels



Règles (de base) pour intégrer des schémas non conflictuels

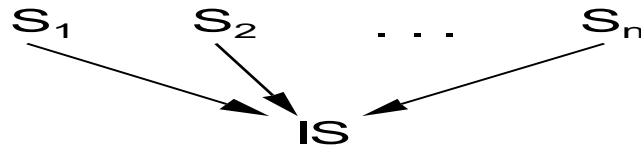


Règles (de base) pour intégrer des schémas non conflictuels

- ▶ Les règles précédentes montrent que 2 schémas non conflictuels sont intégrés par « **superposition** »
- ▶ Mais d'autres règles peuvent s'appliquer dans des cas spéciaux
- ▶ Par exemple, même si $S1.A \text{ DISJOINT } S2.B$, $S1.A$ et $S2.B$ peuvent être représentés par une même classe C couvrant A et B (mais il faut se positionner par rapport à **l'utilité de C car C crée de la confusion**)
- ▶ Une correspondance peut être gardée telle quelle jouant le **rôle de contrainte** sur le schéma intégré
- ▶ Les règles indiquent donc **ce qu'il faudrait** faire dans certains cas, laissant la possibilité d'étendre ces mêmes règles à d'autres cas

Strategies d'intégration

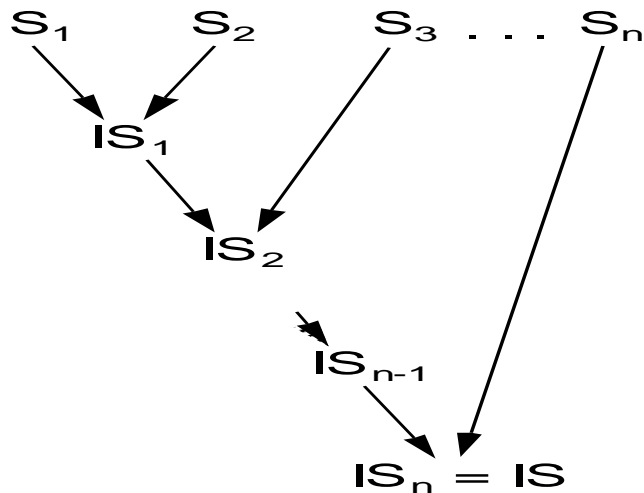
Total (one shot):



all available
local schemas

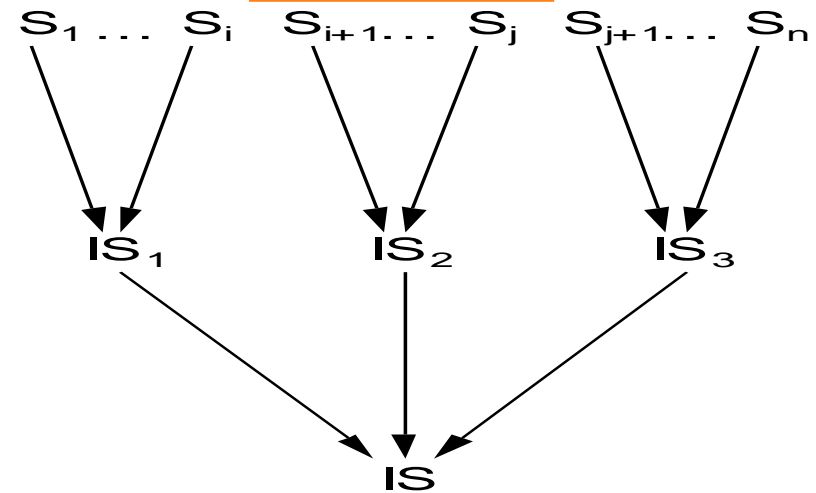
- optimizes the integration process:
 - total knowledge
 - no "rollback" on intermediate choices
- multiplies interschema correspondences

Ladder:



- ordering
- "priority"/"primary" views

Balanced:



IS_1 : sales
 IS_2 : production
 IS_3 : marketing

Intégration de données (ID)

- ▶ L'intégration de données permet un **accès unifié et transparent** à une collection de données mémorisées dans plusieurs **sources autonomes et hétérogènes** [Calvanese 2007]
- ▶ Elle intervient dans plusieurs contextes :
 - Fusion d'entreprises
 - Réorganisation d'entreprise
 - Restructuration de bases de données
 - Combinaison des sources internes et externes à une entreprise (y compris entrepôts)

Intégration de données (ID)

- ▶ Un système, process, flux, outil d'intégration de données fournit un moyen pour obtenir un **état à partir d'états observés**
- ▶ La conception de l'intégration de données se focalise sur l'obtention d'un **état correct** à partir **d'états observés**
- ▶ **La signification d'état correct reste à définir précisément mais, sans doute, il s'agit d'un état sans données redondantes et sans données erronées (sous l'hypothèse que les données d'origine sont correctes)**

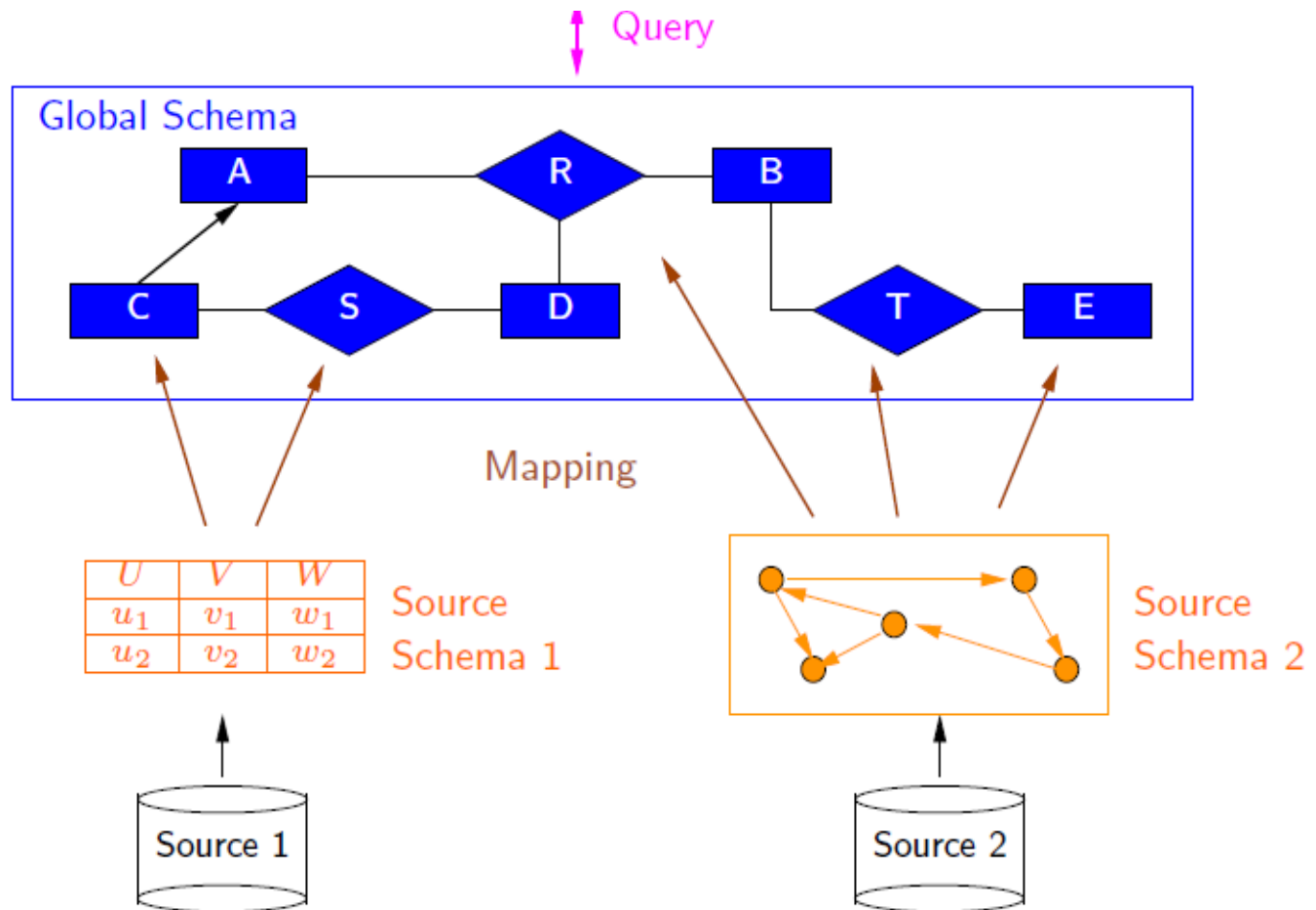
Architectures des systèmes d'ID

Schéma cible,
global ou intégré

Base de
données cible,
globale, ou
intégrée

Schémas
locaux

Bases de
données
locales



Types d'ID

► Bases de données distribuées

Planifiée en conception

- Sources homogènes définies en phase de conception

► Multi bases ou bases fédérées

Proche de la définition de vues

- Sources hétérogènes et autonomes

► Médiateur

Plusieurs types de mappings entre sources et schéma global

- Accès aux sources autonomes et hétérogènes à travers d'un schéma global (*)

► Échange de données généralisé

- Plusieurs sources d'origine et un schéma global cible

► Intégration pair à pair

Sans schéma global (emergent semantics)

- Réseau de sources autonomes et hétérogènes sans schéma global

(*)La notion de schéma global n'est pas la même de celle de schéma intégré

Types de mapping : exemple utilisé

Schéma global

movie(Title, Year, Director)

european(Director)

review(Title, Critique)

Source 1:

r₁(Title, Year, Director) since 1960, european directors

Source 2:

r₂(Title, Critique) since 1990

Informations
additionnelles qui
Ne sont pas
représentables et
Représentées dans les
schémas

Query: Title and critique of movies in 1998

$q(t, r) \leftarrow \text{movie}(t, 1998, d), \text{review}(t, r)$

Requête formulée sur le
schéma intégré

Mapping Global as View (GAV)

Schéma global *movie*(*Title*, *Year*, *Director*)
 european(*Director*)
 review(*Title*, *Critique*)

GAV: to each relation in the global schema, \mathcal{M} associates a **view** over the sources:

$$\begin{array}{lll} q_1(t, y, d) \leftarrow r_1(t, y, d) & \rightsquigarrow & \text{movie}(t, y, d) \\ q_2(d) \leftarrow r_1(t, y, d) & \rightsquigarrow & \text{european}(d) \\ q_3(t, r) \leftarrow r_2(t, r) & \rightsquigarrow & \text{review}(t, r) \end{array}$$

Le mapping \mathcal{M} ci-dessus signifie que
Movie contient (au moins) r_1
European contient (au moins) une projection de r_1
(en pratique le symbole \rightsquigarrow signifie une implication au sens de la logique de 1^{er} ordre)

Notation alternative pour le mapping et les requêtes

- ▶ Notation alternative (extensionnelle) pour un mapping
 - $r1(t,y,d) \subseteq \text{movie}(t,y,d)$ OU
 - *Create view q1(*) as select * from r1 ;*
 - $q1 \subseteq \text{select } * \text{ from movie}$
 - $r1(d) \subseteq \text{european}(d)$ OU
 - *Create view q2 (d) as select d from r1;*
 - $q2 \subseteq \text{select } * \text{ from European}$
- ▶ En notation SQL standard correspondante à une requête sur le schéma intégré
 - *Select t,d from movie inner join review on (movie.t=review.t) where movie.year=1998*

Mapping Local as View (LAV)

Schéma global

movie(Title, Year, Director)

european(Director)

review(Title, Critique)

LAV: to each **source relation**, \mathcal{M} associates a **view** over the global schema:

$r_1(t, y, d) \rightsquigarrow q_1(t, y, d) \leftarrow \text{movie}(t, y, d), \text{european}(d), y \geq 1960$

$r_2(t, r) \rightsquigarrow q_2(t, r) \leftarrow \text{movie}(t, y, d), \text{review}(t, r), y \geq 1990$

Le mapping \mathcal{M} ci-dessus formalise que

r_1 contient **que** films (movies) produits **après 1960** et dont les **directeurs sont européens** (les variables à droite non communes sont existentiellement quantifiées)

LAV nécessite d'une règle de classification plutôt bonne, sinon il est exactement comme le GAV

Notation alternative pour le mapping et les requêtes

- ▶ $r1(t,y,d) \subseteq \{(t,y,d) \mid \text{movie}(t,y,d), y \geq 1960, \text{european}(d)\}$ OU
- ▶ $r1 \subseteq \text{select movie.t, movie.y, movie.d from movie inner join european on (movie.d=european.d) where } y \geq 1960$
- ▶ $r2(t,r) \subseteq \{(t,r) \mid \text{movie}(t,y,d), \text{review}(t,r), y \geq 1990\}$ OU
- ▶ $r2 \subseteq \text{select movie.t,review.r from movie inner join review on (movie.t=review.t) where } y \geq 1990$

Avantages de l'approche LAV

- ▶ LAV permet de représenter une **information incomplète à savoir représentée dans le schéma global mais non disponible dans les sources** (impossible pour GAV)
- ▶ Le schéma global peut donc être conçu d'une manière indépendante des sources ; cela permet d'avoir un **schéma global relativement stable**
- ▶ Les sources peuvent donc être représentées d'une manière plus précise car décrites en fonction d'un schéma global « sémantiquement riche »

Mapping Global Local as View (GLAV)

Schéma global $\text{work}(\text{Person}, \text{Project}), \quad \text{area}(\text{Project}, \text{Field})$

Source 1: $\text{hasjob}(\text{Person}, \text{Field})$

Source 2: $\text{teaches}(\text{Professor}, \text{Course}), \quad \text{in}(\text{Course}, \text{Field})$

Source 3: $\text{get}(\text{Researcher}, \text{Grant}), \quad \text{for}(\text{Grant}, \text{Project})$

GLAV mapping:

$$\begin{array}{ll} q_1^s(r, f) \leftarrow \text{hasjob}(r, f) & \rightsquigarrow q_1^g(r, f) \leftarrow \text{work}(r, p), \text{ area}(p, f) \\ q_2^s(r, f) \leftarrow \text{teaches}(r, c), \text{ in}(c, f) & \rightsquigarrow q_2^g(r, f) \leftarrow \text{work}(r, p), \text{ area}(p, f) \\ q_3^s(r, p) \leftarrow \text{get}(r, g), \text{ for}(g, p) & \rightsquigarrow q_3^g(r, f) \leftarrow \text{work}(r, p) \} \end{array}$$

Base de données canonique

- ▶ La **base de données canonique « simule »** l'existence d'une base de données associée au schéma global
- ▶ Notion facile à mettre en place dans le cas GAV mais plus difficile dans les cas LAV et GLAV
- ▶ Il s'agit d'une notion importante, en particulier, en cas de schéma global « plus riche » que les sources

Exemple : base de données canonique (GLAV)

Consider $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with

Global schema \mathcal{G} : $\text{student}(\text{Code}, \text{Name}, \text{City})$
 $\text{enrolled}(\text{Scode}, \text{Ucode})$

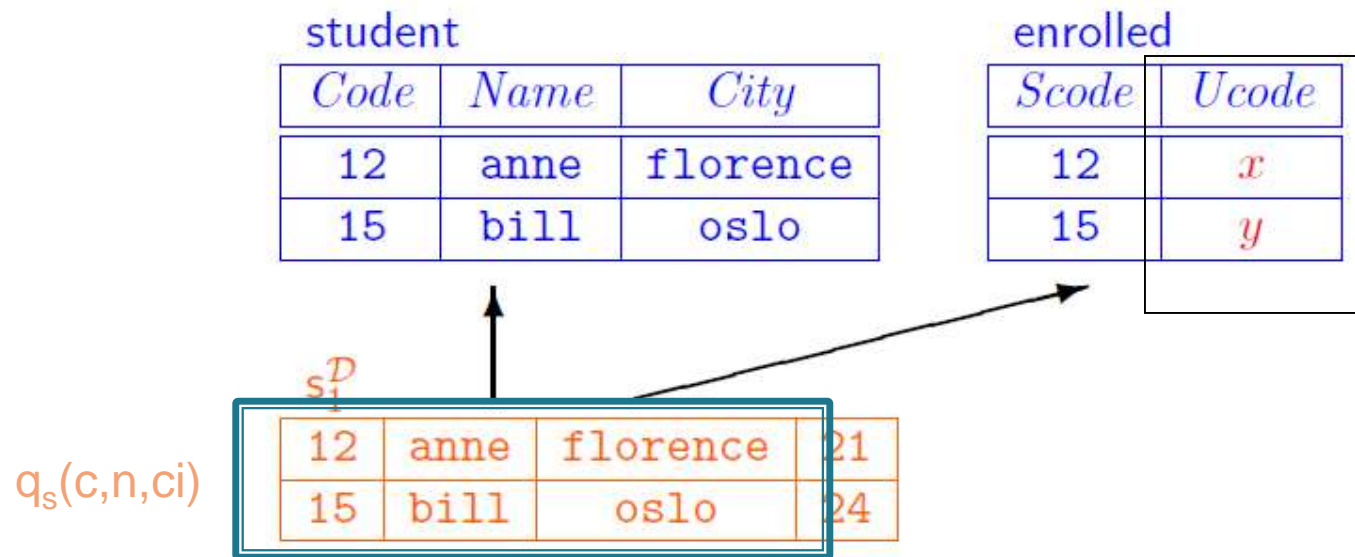
Source schema \mathcal{S} : relation $s_1(\text{Scode}, \text{Sname}, \text{City}, \text{Age})$

Mapping \mathcal{M} :

$$q_s(c, n, ci) \leftarrow s_1(c, n, ci, a) \quad \rightsquigarrow \quad q_g(c, n, ci) \leftarrow \begin{array}{l} \text{student}(c, n, ci), \\ \text{enrolled}(c, u) \end{array}$$

Exemple : base de données canonique (LAV)

$$q_s(c, n, ci) \leftarrow s_1(c, n, ci, a) \rightsquigarrow q_g(c, n, ci) \leftarrow \text{student}(c, n, ci), \text{enrolled}(c, u)$$



Des valeurs doivent exister : il suffit de les créer via “des variables” toujours distinctes

A source db \mathcal{D} and a corresponding possible global db.

Règle inversée

$$\forall x,y [S1(x,y) \rightarrow \exists z [e(z,x) \wedge r(z,y)]]$$

$$\equiv \forall x,y \neg S1(x,y) \vee [e(\mathbf{f(x,y)},x) \wedge r(\mathbf{f(x,y)},y)] \quad \textit{Skolem terms}$$

$$\equiv \forall x,y \quad [\neg S1(x,y) \vee e(\mathbf{f(x,y)},x)] \wedge \\ [\neg S1(x,y) \vee r(\mathbf{f(x,y)},y)]$$

$$\equiv \forall x,y \quad [S1(x,y) \rightarrow e(\mathbf{f(x,y)},x)] \wedge \\ [S1(x,y) \rightarrow r(\mathbf{f(x,y)},y)]$$

$$S1(x,y) \approx> e(\mathbf{f(x,y)},x), S1(x,y) \approx> r(\mathbf{f(x,y)},y)]$$

Exemple d'application de la règle inversée

$$q_s(c, n, ci) \leftarrow s_1(c, n, ci, a) \rightsquigarrow q_g(c, n, ci) \leftarrow \text{student}(c, n, ci), \text{enrolled}(c, u)$$

student

<i>Code</i>	<i>Name</i>	<i>City</i>
12	anne	florence
15	bill	oslo

enrolled

<i>Scode</i>	<i>Ucode</i>
12	<i>x</i>
15	<i>y</i>

f(12,anne,florence)

f(15,bill,oslo)

$s_1^{\mathcal{D}}$

12	anne	florence	21
15	bill	oslo	24

A source db \mathcal{D} and a corresponding possible global db.

Autres problématiques

- ▶ Prise en compte de données dans les sources : Extraction de données, nettoyage, réconciliation des conflits entre données, en particulier
 - Identification des entités (record linkage, dé-doublonnage etc.)
- ▶ Découverte de mappings
- ▶ Propagation des modifications (proche du « view update »)

Identification des objets

Teaching_Personnel

<u>Name</u>	<u>Surname</u>	Street	number
Diego	Milani	HebelStrasse	90

Font ils référence à la même personne ?

Teaching_Personnel

<u>Name</u>	<u>Surname</u>	Street	number
Diego	Milani	HebelStrasse	90

Erreurs locales à résoudre

Teaching_Personnel

<u>Name</u>	<u>Surname</u>	Address	Sex
Diego	Milani	HebelStrasse 90	F
Heiko	Schuldt	...	M

Standardisation nécessaire

Personnel

<u>Surname</u>	Address	T
Milani	HebelStrasse 90	d
Schuldt	MittlerStraße 90	P

Difficulté de l'identification des objets

S1: Teaching_Personnel

<u>ID</u>	Name	Surname	Address
DM	Diego	Milano	HebelStrasse 90
HS	Heiko	Schuldt	...

S1: Teaching_Personnel

<u>ID</u>	Name	Surname	Address
06	Diego	Milano	HebelStrasse 90
02	Heiko	Schuldt	...

Est il possible que des identifiants différents correspondent aux « mêmes choses » ?

S1: Teaching_Personnel

<u>Name</u>	<u>Surname</u>	Address
Diego	Milano	HebelStrasse 90
Heiko	Schuldt	...

S2: Teaching_Personnel

<u>Name</u>	<u>Surname</u>	Address
Diego	Milano	MissionStrasse 23
Heiko	Schuldt	...

Histoires différents ou objets différents ?

Difficulté de l'identification des objets

Signification de valeurs

Ville	Vente	Produit	Quantité
Vannes		P&Q	



Est-ce qu'il s'agit de
la ville
« administrative » ?



Est-ce qu'il s'agit
d'une fusion de
plusieurs produits ?

Découverte de mappings

- ▶ Etant donné 2 ou plusieurs sources, leurs schémas, et un schéma global, comment trouver
 - Des mappings (requêtes) liant le schéma global avec les schémas locaux
 - Des mappings (requêtes) liant les schémas locaux (dans le cas pair à pair)
- ▶ Est ce qu'il y a une **méthode semi-automatique** pour trouver des **mappings respectant certaines propriétés** ? Quelles informations sont nécessaire pour trouver ces mappings ?
- ▶ Qu'est ce qui se passe en cas de **superposition des sources** ?

Propagation des modifications

- ▶ Lorsqu'un mapping est défini, si ce mapping contient une **fonction d'agrégation**, il sera probablement impossible de propager une modification sur la base globale vers des modifications des bases locales
- ▶ Ce problème est critique si des modifications à effectuer sur les données sont prévues (mais dans les entrepôts il n'y a pas de modifications faites au niveau de l'entrepôt même ou du schéma intégré)

ID et ETL

- ▶ Les ETL sont des outils permettant de mettre en place un **chargement de données** dans un entrepôt, à partir de plusieurs sources de données (hétérogènes)
- ▶ Les **ETL permettent la conception des processus de chargement**, étant chaque processus une séquence d'opérations sur les données sources, sur des données intermédiaires, sur les données cibles
- ▶ Les opérations principales restent des **requêtes pour mettre en correspondance les données sources et les données cibles**, tout en tenant compte des plusieurs possibilités (types GAV, LAV, GLAV) et transformations ; **mais un processus ETL n'est pas une requête « complexe » ; plutôt un processus ETL est une spécification exécutable d'un mapping entre schémas**

ID et IS

- ▶ L'intégration de schémas **peut** fournir un schéma global pour une ID
- ▶ Par contre, un schéma intégré est forcément **assez dépendant** des schémas locaux (des sources) ; pour cela, un **schéma intégré pourrait se révéler « instable »** vis-à-vis des modifications des schémas locaux ou de la prise en compte des nouveaux schémas locaux
- ▶ L'intégration de schémas fournit généralement un **mapping type GAV à valider**
- ▶ **Cependant ce mapping GAV reste « théorique » car se basant sur des correspondances inter-schéma qui ne sont que des hypothèses**

Mappings GAV théoriques

- ▶ Si A est un unique artefact du schéma intégré qui remplace **entièrement** A_1, \dots, A_n alors un mapping GAV (théorique, à savoir sans prendre en compte la qualité de données) a la forme suivante :
 - $A \supseteq \Pi A_1 \cup \dots \cup \Pi A_n$
 - Sous l'hypothèse de transformer les identifiants suivant les WCI et les autres propriétés suivant les WCP (à l'aide de Π)
- ▶ Le but du mapping étant ce représenter un **état représentatif** des états des sources ; en général, cela veut dire un état contenant toutes les données et pour les données correspondantes, l'état le plus correct, le plus complet, le plus récent
- ▶ Il n'est pas certain que tout mapping puisse s'exprimer d'une manière simple à travers un mapping GAV

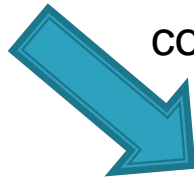
Exemple

Client
nom
email

S1.Contact CONTAIN S2.Client
WCI : Client.email=Contact.email
WCP : Client.nom=Contact.nom

Contact
nom
prénom
email

Règles solution
conflits + intégration
schémas non
conflictuels



INT_Client
nom
prénom
email

Point de vigilance : Les règles de solution de conflits et d'intégration de schémas non conflictuels ne représentent que, pour un client/contact, il n'y a qu'un seul email et un seul nom pour un même email (email est unique et un seul attribut nom dans le schéma intégré)

Mapping GAV théorique :

INT_client \supseteq C1(email, *null*, nom) UNION C2(email, prénom, nom)

Aperçu mappings GAV ajustés

- ▶ Pour alimenter **INT_client** il faut donc utiliser le contenu des tables qui correspondent à savoir **Client** et **Contact**
- ▶ S'il s'agit que de 2 tables **C1 / C2**, pas mal de possibilités se présentent (pistes) :
 - **INT_client** \supseteq **std-net**(C1) UNION **std-net**(C2) /* le contenu doit être standardisé et nettoyé pour tous les attributs ; std-net peut avoir déjà été appliqué lors de l'échange de données ; std-net peut se baser sur une analyse/exploration de données contextuelle
 - **INT_client** \supseteq C1 /* utilise une source master
 - **INT_client** \supseteq select **std-net**(x.email),.... from (C1 union ALL C2) x group by **std-net**(x.email),.... /* le contenu doit être standardisé et nettoyé
 - **INT_client** \supseteq C1 UNION ALL C2 /* aucune fusion de données
 - **INT_client** \supseteq C1 left join C2 on std(C1.email) = std(C2.email) /* le même client est celui ayant le même email mais il est possible de rajouter d'autres attributs
 - **INT_client** \supseteq C1 left join C2 on std-net(C1.email) = std-net(C2.email) UNION C2 left join C1 on std-net(C1.email) = std-net(C2.email) /* il faut décider d'où proviennent les attributs non égalisés et en commun
 - **INT_client** \supseteq Select x+y from Select x, y from C1 x, C2 y where **sim**(x,y) > **seuil**
UNION Select x from C1 x where x not in (Select x from C1 x, C2 y where **sim**(x,y) > **seuil**)
UNION Select y from C2 where y not in (Select y from C1 x, C2 y where **sim**(x,y) > **seuil**)
/* utilise une similarité **sim**(x,y) et une fusion commutative + pour les attributs en commun et **null** pour tout attribut non commun
 - **INT_client** \supseteq
select x+y from
 - (Select x, **max(sim)** as **maxsim** from
 - C1 z left join /* best match similarity per C1 row
 - (Select x, y, **sim**(x,y) as **sim** from C1 x, C2 y where **sim**(x,y) > **seuil**) on (x=z)
 - group by x),
 - left join
 - (Select x1, y, **sim**(x,y) as **sim** from C1 x, C2 y where **sim**(x,y) > **seuil**) on (x=x1)
where **sim**=**maxsim** /* best match(es)

Voir aussi articles et documents sur la détection de doublons, d'entité, record matching, string similarity

Problématiques mappings GAV ajustés

- ▶ Les mappings peuvent ne pas aboutir sur un résultat acceptable (mauvaise qualité des identifiants notamment, standardisation impossible), la correspondance ne peut pas être utilisée directement et doit être interprétée comme « cible » (plus comme mapping LAV)
- ▶ Dans ce cas, il faut d'abord voir la correspondance comme moyen pour détecter des doublons dont le résultat doit être validé avant d'être mis en œuvre
 - Tous les mappings GAV ajustés possibles sont en principe appliqués
 - Utilisation du WCP à la place du WCI pour générer le mapping
 - Le contenu de INT_Client doit être validé
 - Ces mappings ne prennent pas en compte les intégrations successives pour lesquelles un problème d'intégration de données se pose entre les sources et les données précédemment intégrées

Aperçu de sim (chaines de caractères)

- ▶ $\text{sim}(x,y) = \text{simlin}(x,y) = 0.2s_{\text{nom}}(x,y) + 0.3s_{\text{email}}(x,y) + \dots$
- ▶ $\text{sim}(x,y) = 1 / (1 + e^{-\text{simlin}(x,y)})$
 - $s_{\text{nom}}(x,y)$: Jaro–Winkler
 - $s_{\text{email}}(x,y)$: edit distance
 - ...
 - $\text{JaroWinklerSim}(s1, s2) = \text{JaroSim}(s1, s2) + |\rho| \times f \times (1 - \text{JaroSim}(s1, s2))$
 - $\text{JaroSim} = 1 / 3 \times (|\sigma| / |s1| + |\sigma| / |s2| + (|\sigma| - 0.5t) / |\sigma|)$
 - $\text{LevDist}(s1, s2)$ = the minimum number of character insertions, deletions, and replacements necessary to transform $s1$ to $s2$.
- ▶ utl_match package ORACLE PL/SQL

Apprentissage ?

- ▶ Non supervisé

- Clustering avec « sim » (“create model” in SQL)

- ▶ Supervisé

- Feature vector avec différentes similarités par exemple

$$v = \langle s_1(x,y), \dots, s_m(x,y) \rangle$$

- Training set $\langle v, l \rangle$ $l=0/1$, yes/no

- ▶ Combinaison de plusieurs modèles

Monitoring (ETL)

- ▶ La difficulté de mettre en place les mappings GAV demande un monitoring des données traitées par l'ETL
- ▶ En particulier, il faudra inspecter les différences entre les données d'entrée, provenant des sources, et les données transformées insérées dans la cible (schéma intégré)
- ▶ Cela peut être aussi fait avec la mise en place de contraintes sur le schéma intégré

Exemples simples de monitoring

- ▶ Comptage données d'entrée, données insérées dans le schéma intégré
 - `Select count(*) from Client(1)`
 - `Select count(*) from INT_Client`
- ▶ Jointures
 - `Select Client(1) from Client(1) left join INT_Client on Client(1).ID=INT_Client.ID /* qui sont les clients de la source 1 qui n'ont pas été insérés dans INT_Client`
- ▶ Données rejetées par de contraintes non satisfaites
 - Contrainte : tous les clients doivent avoir une adresse email valide
 - `Select * from INT_Client where email is not valide`

Perspective procedurale de l'approche "supply-driven"

