Bases de Données relationnelles

Télécom Nancy 1ère année

Malika SMAÏL-TABBONE

Maître de conférences

(malika@loria.fr)

Plan du cours (1/3)

Chap. 1 : Introduction

- Limites des systèmes de fichiers
- Notion de BD et de SGBD
- Architecture et fonctions d'un SGBD
- Processus de conception d'une BD

Chap. 2 : Modèle Entité-Association

- Concepts de base du modèle E/A
- Règles de complétude d'un schéma E/A

Plan du cours (2/3)

Chap. 3 : Modèle relationnel de données

- Concepts du modèle relationnel
- Transformation d'un modèle entité-association en modèle relationnel
- Redondance et normalisation
- Langages de manipulation des données (LMD)

Chap. 4 : Le langage SQL (*SGBD ORACLE*)

- Définition/Mise à jour/Interrogation des données
- Les vues
- Catalogue du système ORACLE (dictionnaire)

Plan du cours (3/3)

Chap. 5 : PL/SQL et Triggers

Chap. 6 Gestion des transactions

Mini-bibliographie : BD relationnelles et systèmes d'information

1. « Fundamentals of database systems » R. Elmasri et S. B. Navathe, troisième édition, The Benjamin/Cummings Pub., 2000.
2. « Bases de données et systèmes d'information » N. Boudjlida, Dunod, 1999.
3. « Apprendre et pratiquer MERISE » J. Gabay, Masson, 1989.
4. « Bases de données : objet et relationnel » G. Gardarin, Eyrolles, 1999.
4. « Bases de données : les systèmes et leurs langages » G. Gardarin, Eyrolles, 1984.

Chap. 3 : Modèle relationnel de données

1. Concepts du modèle relationnel

2. Transformation d'un modèle entité-association en modèle relationnel

3. Redondance des données et normalisation

4. Langages de manipulation des données (LMD)

    a- Algèbre relationnelle (AR)

    b- Calcul relationnel de tuples (CRT)

Modèle relationnel : un peu d'histoire

E.F. Codd dans *CACM 1970*

« A Relational Model of Data for Large Shared Databanks »

Le modèle relationnel est un **modèle logique** associé aux SGBD relationnels (cf. processus de conception de BD)

Base de données vue par l'utilisateur =

Ensemble de **tableaux** (ou de **tables**)

Nombreux travaux fondamentaux sur les :

- méthodes de conception de BDR
- langages de manipulation des données

Modèle relationnel : un peu d'histoire

SGBD relationnels

1976 : Premières réalisations (SYSTEM-R, INGRES)

1980 : Premières commercialisations

2003 : marché inondé (Oracle, Sybase, Informix, MS Access…)

Modèle relationnel : deux parties « théoriques »

- Concepts du modèle (*table, attribut, domaine* …)
- Langage de manipulation des données
    - langage algébrique ou algèbre relationnelle
    - langage prédicatif (formules du CP1) : CRT

3.1. Concepts du modèle relationnel
a- Relations, attributs et tuples

- Une **relation** est caractérisée par :
  - un nom R
  - un ensemble d'attributs A_1, A_2, … A_n

- Si une relation a ***n*** attributs, ***n*** est son **arité**.
- Notation d'une relation

R (A_1, A_2, … A_n)

ex. **`Produit (numProd, libellé, pu)`**

a- Relations, attributs et tuples

- Un **attribut** est caractérisé par :
  - un nom A_i
  - un domaine noté **dom(A_i)**, ensemble des valeurs possibles de A_i

    ex. : *dom (prix) =]0,10000]*

- Valeur **nulle** (notée **`NULL`**) : valeur particulière indiquant que la valeur d'un attribut (non obligatoire) n'est pas connue ou que l'attribut ne s'applique pas.

  ex1. Cas un client dont on ignore la ddn.

  ex2. Cas d'un employé ne possédant pas de téléphone.

a- Relations, attributs et tuples

- Un **tuple** d'une relation R (A_1, A_2, … A_n) est un ensemble de valeurs **<v_1, v_2 , …v_n>** telles que

  $v_i \in dom(A_i)$

  ou

  v_i a la valeur NULL

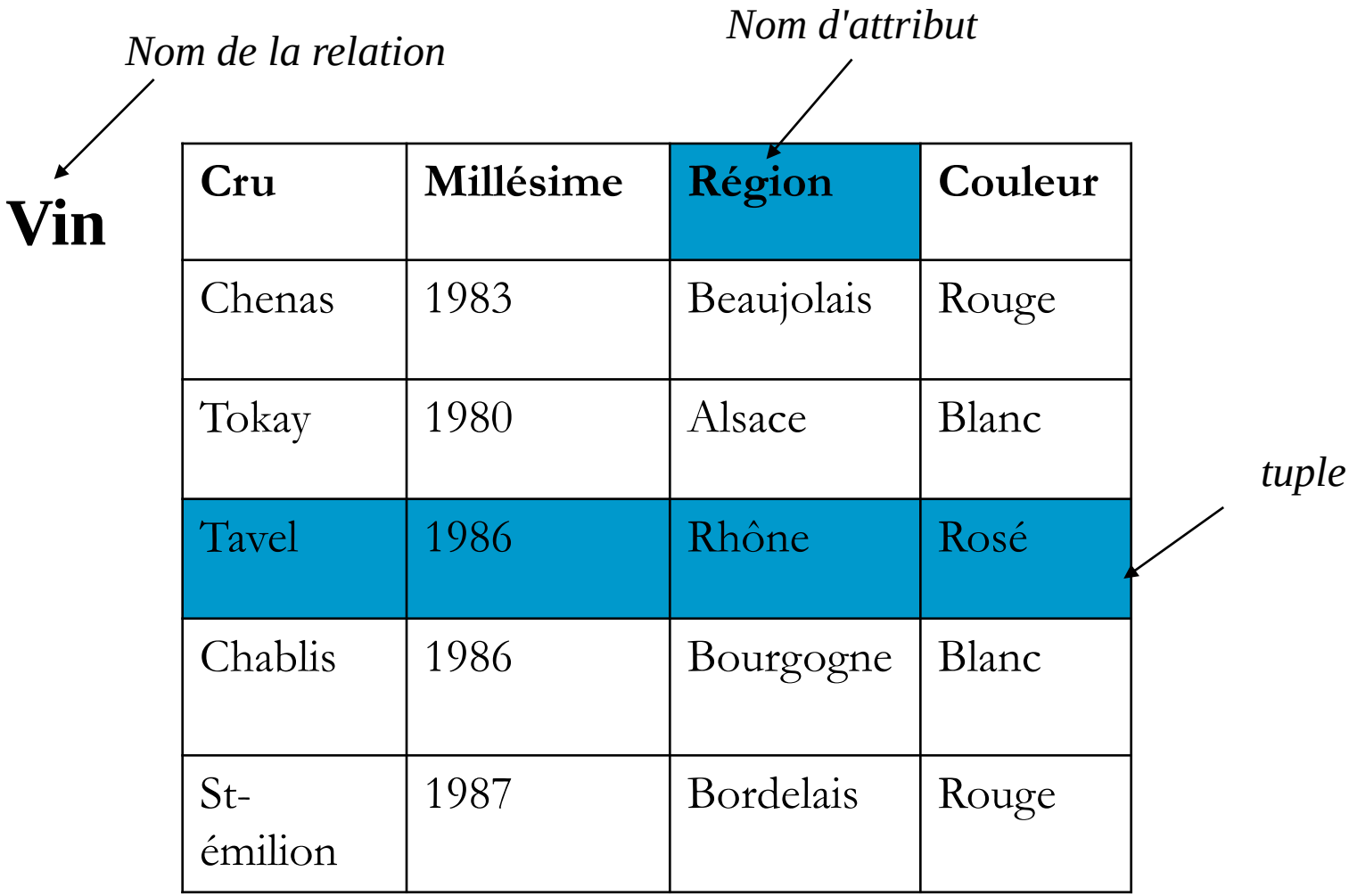
ex. *<36, écrou, 5> est un tuple de la relation Produit*.

a- Relations, attributs et tuples

L'**extension** d'une relation

- est l'ensemble de ses tuples
- est représentée par un **tableau** à deux dimensions
  - une **colonne** correspond à un **attribut**
  - une **ligne** correspond à un **tuple**

Extension d'une relation Vin



Cru	Millésime	Région	Couleur
Chenas	1983	Beaujolais	Rouge
Tokay	1980	Alsace	Blanc
Tavel	1986	Rhône	Rosé
Chablis	1986	Bourgogne	Blanc
St-émilion	1987	Bordelais	Rouge

Différents types d'attributs

- Attribut **atomique** versus **composé** (subdivisé en attributs)

  ex. les attribut *nom* et *prix* sont atomiques

  ex. l'attribut *adresse* est composé de trois attributs : *rue, ville, CP*

- Attribut **dérivé** : calculé à partir d'autre(s) attribut(s)

  ex. attribut *âge* calculé à partir *de l'attribut date_de_naissance*

- Attribut **monovalué** versus **multivalué** (plusieurs valeurs par tuple)

  ex. *nom* versus *prénoms_des_enfants*

Notion de base de données relationnelle(s)

- Une **base de données relationnelle** est un ensemble de relations liées entre elles

ex. La BD de Vins formée des trois relations

```
Buveur (nob,nom,prénom)
Vin (nov, cru, millésime,degré,couleur)
Consommation (nob,nov,date,quantité)
```

a- Relations, attributs et tuples

Le schéma d'une relation est défini par :

- ✓ le nom de la relation
- ✓ la liste des attributs + domaines
- ✓ des contraintes d'intégrité

ex. `Produit(numProd : nombre entier,`
`libellé : chaîne de caractères,`
`pu : nombre réel)`

Deux contraintes :

1) Un produit est identifié par `numProd(numProd :` clé primaire)
2) `0 < pu ≤ 10 000`

b- Contraintes d'intégrité (CI)

- **Clé d'une relation** : Groupe d'attributs **minimum** qui identifie de manière **unique** un tuple dans toute extension de la relation
  - cf. Notion d'identifiant de type d'entité dans modèle E-A

    ex : {cru, millésime,couleur} dans la relation Vins

    {no-sécu} dans une relation Personne.

- Toute relation doit avoir au moins une clé, c'est la **clé primaire**

  ex. numProd : clé primaire de Produit.

  {cru, millésime, couleur} : clé primaire de Vin.

- Notation : la clé primaire est **soulignée** dans le schéma.

  ex. Produit (<u>numProd</u>, libellé, pu)

b- Contraintes d'intégrité (CI)

- **Clé étrangère d'une relation** : attribut(s) constituant la clé primaire d'une autre relation (ou dont la valeur est unique dans l'autre relation)

- Les clés étrangères définissent les CI référentielles

- Notation : la clé étrangère est en **italique** dans le schéma.

ex. Soit la base de données :

```
Buveur (nb,nom,prénom)
Vin (nv, cru, millésime,degré)
Abus (nb,nv,date,quantité)
```

Abus.nv est une clé étrangère, référençant *Vin.nv*
Abus.nb est une clé étrangère, référençant *Buveur.nb*

b- Contraintes d'intégrité

- **Contraintes liées au domaine** : les données doivent vérifier certaines conditions pour être cohérentes.

ex. `pu > 0  ET  pu ≤10000`

`millésime > 1900`

`date_fin_projet > date_début_projet`

3.2. Transformation d'un modèle entité-association en modèle relationnel

Transformation d'un schéma E/A en un schéma relationnel en 6 étapes :

1) Chaque **entité** donne une relation de même nom.

- Les attributs de l'entité deviennent attributs de la relation
- Seuls les attributs simples des attributs composés sont inclus
- L'identifiant de l'entité devient clé primaire de la relation

ex. Relation obtenue à partir de l'entité Client (exemple VPC)

`Client (numCli,nom,prénom,ddn,rue,CP,ville)`

3.2. Transformation d'un modèle entité-association en modèle relationnel

Chaque **entité faible** I, donne une relation R qui comprend tous les attributs de I ainsi que la clé de l'entité identifiante

- La clé de R est la concaténation de la clé partielle de I et de la clé de l'entité identifiante.

ex. Cas de l'entité faible **Exemplaire**

Ouvrage (numO, titre, auteur, éditeur)
Exemplaire (*numO*, numE, position, date-achat)

3.2. Transformation d'un modèle entité-association en modèle relationnel

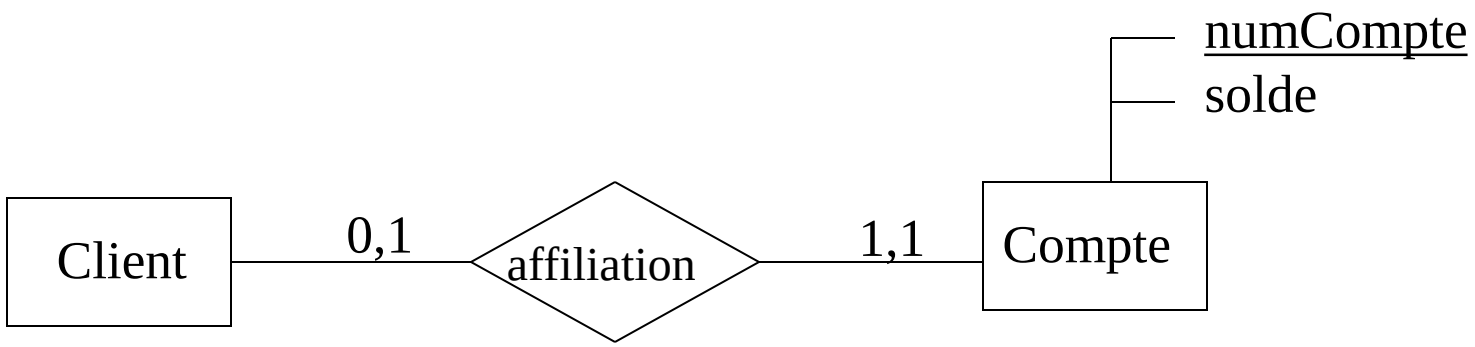
2) **C**haque **association de type 1-1** est prise en compte en incluant la clé primaire d'une des relations comme clé étrangère dans l'autre relation

- idem pour les attributs éventuels de l'association

Si les cardinalités minimales sont non nulles, il est possible de :

- fusionner les deux relations
- y ajouter les attributs de l'association
- choisir comme clé une des deux clés

ex. Dans l'exemple VPC, si un client peut posséder un compte :



Cela donne 3 possibilités :

Compte (numCompte,solde)
Client (numCli,nom,prénom,ddn,rue,CP,ville,***numCompte***)

Ou alors :

Compte (numCompte,solde,***numCli***)
Client (numCli,nom,prénom,ddn,rue,CP,ville)

Ou alors :

Client (numCli,nom,prénom,ddn,rue,CP,ville,***numCompte,solde***)

3.2. Transformation d'un modèle entité-association en modèle relationnel

3) Chaque **association de type 1-N** est prise en compte en incluant la clé primaire de la relation de cardinalité N comme clé étrangère dans l'autre relation

   - idem pour les attributs éventuels de l'association

ex. Cas de l'association **fournit**

`Fournisseur (NumFour, raisonSoc)`

`Produit (numProd,, libellé, pu, `***`numFour`***`)`

3.2. Transformation d'un modèle entité-association en modèle relationnel

4) Chaque **association de type N-M** donne lieu à une nouvelle relation incluant :

- les clés primaires des relations participantes
- les attributs de l'association

Puis choix d'une clé minimale parmi ces attributs

ex. Cas de l'association **Commande**

`Commande (`***`numCli`***`, `***`numProd`***`, `**`date`**`, quantité)`

3.2. Transformation d'un modèle entité-association en modèle relationnel

5) Chaque **association de degré supérieur à 2** donne lieu à une relation ayant comme attributs la liste des clés primaires des entités participantes et la liste des attributs éventuels de l'association.

- puis choix d'une clé minimale parmi ces attributs.

6) Chaque **attribut dérivé** donne lieu à une **vue** SQL dont la définition correspond à la traduction en SQL de la règle de dérivation.

Schéma relationnel complet de l'exemple VPC

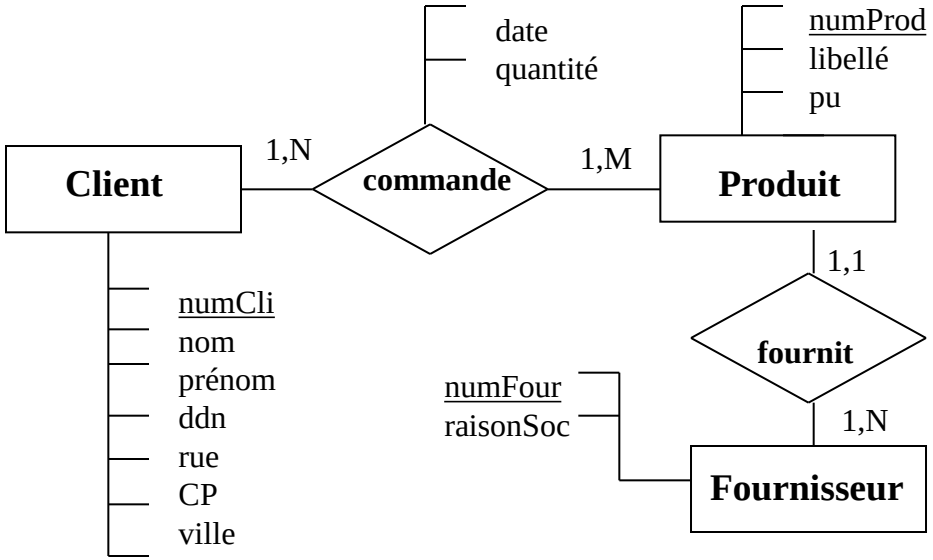


Schéma relationnel

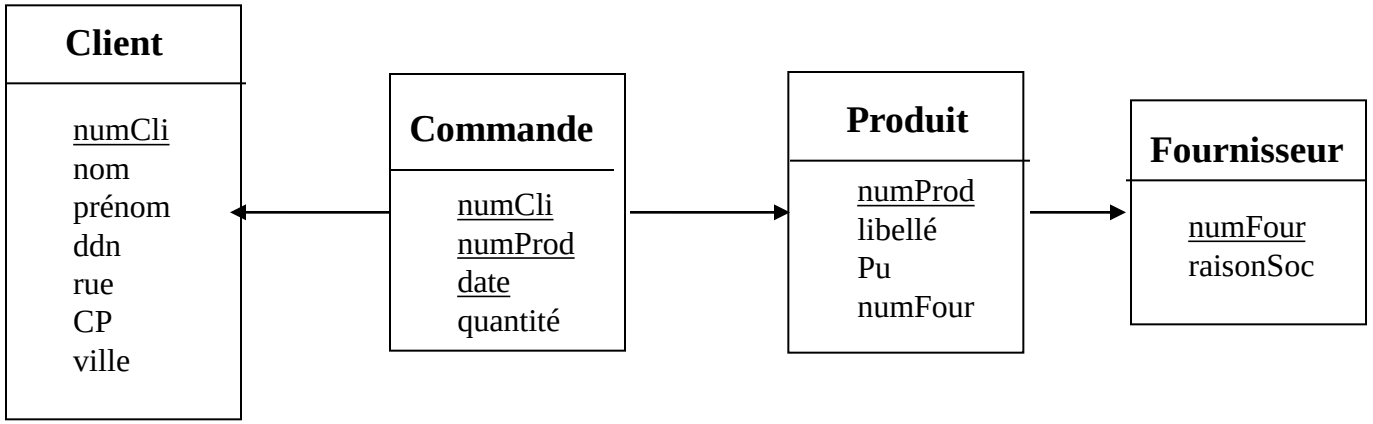
Client (numCli,nom,prénom,ddn,rue,CP,ville)

Produit (numProd, libellé, pu, ***numFour***)

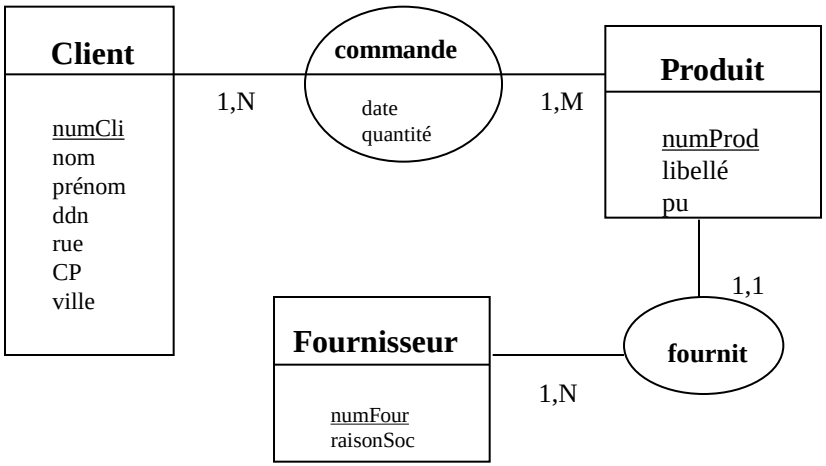
Fournisseur (numFour, raisonSoc)

Commande (***numCli***, ***numProd***, **date**, quantité)

Représentation graphique du modèle relationnel



à ne pas confondre avec le modèle E/A !



A vous : Effectuer la traduction E/A → relationnel

Cas 1 : mini-bibliothèque

Schémas des relations obtenues

Ouvrage (numO,titre,auteur,éditeur)

Mot-clé (numM,sujet)

Exemplaire (*numO*,numE,position,date-achat, *numL*)

Lecteur (numL,nom,adresse)

Caractérise (*numO*,*numM*)

Exemple de matérialisation d'une association N-M

Ouvrage

NumO	Titre	Auteur	Editeur
1	La bête humaine	Hugo	Livre de Poche
2	Gaston Lagaffe – 3	Franquin	Dupuis

Mot-clé

numM	sujet
1	Drame
2	Réalisme
3	BD

Caractérise

numO	numM
1	1
1	2
2	3

Chap. 3 : Modèle relationnel de données

1. Concepts du modèle relationnel

2. Transformation d'un modèle entité-association en modèle relationnel

3. Redondance des données et normalisation

4. Langages de manipulation des données (LMD)

    a- Algèbre relationnelle (AR)

    b- Calcul relationnel de tuples (CRT)

3.3. Redondance de données et normalisation

Exemple d'une gestion de stocks : des produits sont stockés dans des dépôts dans certaines quantités

- Soit une relation simple dont le schéma est :

Produit (numP, libelléP, puP, numD, qté, adrD, volumeD)

- et son extension :

numP	libelléP	puP	numD	qté	adrD	volumeD
P1	K7	24	d1	300	Nancy	9000
P1	K7	24	d2	500	Laxou	6000
P2	Vis12	0.2	d4	900	Nancy	20000

3.3. Redondance de données et normalisation

On constate :

- que la relation est **redondante**
- qu'il y a **risque d'introduction d'incohérence**
- qu'il y a **risque de perte d'information**

Normalisation = moyen d'éliminer la redondance à double but :

- suppression des problèmes de mise à jour
- réduction de l'espace de stockage

3.3. Redondance de données et normalisation

Les dépendances fonctionnelles (DF)

Soit R(X,Y,Z) une relation ou X,Y,Z sont des ensembles d'attributs. Z peut être vide.

Définition :

L'ensemble d'attributs Y dépend fonctionnellement de l'ensemble d'attributs X dans la relation R si la valeur de X détermine la valeur de Y dans toute extension de R.

Notation : $\mathbf{X \rightarrow Y}$

Ex. Produit (numProd, libellé, pu)

numProd → libellé *le numéro d'un produit détermine son libellé*

numProd → pu *e numéro d'un produit détermine son prix unitaire*

3.3. Redondance de données et normalisation

Dépendances fonctionnelles et clé de relation

Soit R(A_1, A_2, ..., A_n) un schéma de relation, et X un sous-ensemble de (A_1, A_2, ..., A_n)

X est une clé de R si et seulement si :

- $X \rightarrow (A_1, A_2, ..., A_n)$
- et X est minimal

3.3. Redondance de données et normalisation

Propriétés des DF

Réfexivité : Si $Y \subseteq X$ alors $X \rightarrow Y$

Augmentation : si $W \subseteq Z$ et $X \rightarrow Y$ alors $X,Z \rightarrow Y,W$

Transitivité : si $X \rightarrow Y$ et $Y \rightarrow Z$ alors $X \rightarrow Z$

Pseudo-transitivité : si $X \rightarrow Y$ et $Y,Z \rightarrow T$ alors $X,Z \rightarrow T$

Union : si $X \rightarrow Y$ et $X \rightarrow Z$ alors $X \rightarrow Y,Z$

Décomposition : si $Z \subseteq Y$ et $X \rightarrow Y$ alors $X \rightarrow Z$

Précision : X,Y est équivalent à $X \cup Y$

Les formes normales

Première forme normale (1NF)

Une relation est en 1NF si chacun de ses attributs a un domaine ***atomique*** *(non décomposable) et* ***monovalué****.*

Ex1. La relation

Personne (nom, prénoms, âge)

n'est pas en 1NF si l'attribut prénoms peut avoir cette valeur : {julien, marc, christophe}

Les formes normales

Ex2. La relation

Personne (nom, prénom, adresse)

n'est pas en 1NF si l'attribut *adresse* se décompose en *rue*, *CP*, *ville*.

Les formes normales

Deuxième forme normale (2NF)

Une relation est en 2NF si :

- elle est en 1NF

- et tout attribut n'appartenant pas à la clé ne dépend pas d'une partie de la clé.

Ex. **Client(numCli, nom, prénom, ddn, rue, CP, ville)**

la relation Client est en 2NF

Les formes normales

Deuxième forme normale (2NF)

Stock (<u>numProd, numDép</u>, libelléProd, qtéStockée)

La relation Stock n'est pas en 2NF car

numProd→libelléProd

Stock peut être décomposée en deux relations en 2NF :

Produit (<u>numProd</u>, libelléProd)

Stock2 (<u>numProd, numDép</u>, qtéStockée)

Les formes normales

Troisième forme normale (3NF)

Une relation est en 3NF si :

- *elle est en 2NF*
- *et il n'existe aucune DF entre attributs non clé.*

Les formes normales

Troisième forme normale (3FN)

Ex. La relation

Avion (no-avion, constructeur, type, capacité)

n'est pas en 3NF puisque

type → constructeur et type → capacité

Décomposition en deux relations en 3NF

Avion2 (no-avion, type)

Modèle (type, constructeur, capacité)

Décomposition d'une relation

- Toute relation a au moins une décomposition en 3NF qui est sans perte d'information* et qui préserve** les dépendances fonctionnelles.

* : *la relation initiale est obtenue par jointure naturelle des relations issues de sa décomposition.*

** : *pour chaque DF, au moins une relation issue de la décomposition contient les parties gauche et droite de la DF*

- **Théorème de Heath :**

Une relation **R(A,B,C)** telle que **A → B** est égale à la jointure de ses projections sur **(A,B)** et **(A,C)**

La décomposition de R en deux relations **R1** (**A**,**B**) et **R2** (**A**,**C**) préserve la DF **A → B** et est sans perte d'information

Démarche naïve pour la normalisation d'une relation

1. Identifier les DF
2. Choisir une DF et appliquer le théorème de Heath pour décomposer la relation (qui n'est pas en 3NF)
3. Pour chaque relation issue de la décomposition qui n'est pas en 3NF, itérer à 2.

Le processus aboutit à une liste de relations en 3NF (décomposition sans perte d'information) mais ***certaines DF peuvent être perdues***

Exemple de normalisation(1/3)

Soit la relation :

FILM (no-exploitation, titre, réalisateur)

et les D.F. suivantes :

no-exploitation $\rightarrow$ titre

titre $\rightarrow$ réalisateur

Sous quelle forme normale est cette relation ?

Exemple de normalisation (2/3)

1NF ? Oui, car tous les attributs sont simples.

2NF ? Oui, car si on choisit {no-exploitation} comme clé primaire, tous les attributs non clé dépendent de la clé.

DF1 : no-exploitation $\rightarrow$ titre

DF2 : titre $\rightarrow$ réalisateur

et par transitivité :

DF3 : no-exploitation $\rightarrow$réalisateur.

3NF ? **Non**, car DF2 est une D.F. entre deux attributs non clé

Exemple de normalisation (3/3)

Décomposition en utilisant DF2

FILM1 (<u>no-exploitation</u>, *titre*)

FILM2 (<u>titre</u>, réalisateur)

- Ces deux relations sont en 3NF et toutes les D.F. sont conservées.

- FILM = JOIN(FILM1,FILM2)

Chap. 3 : Modèle relationnel de données

1. Concepts du modèle relationnel

2. Transformation d'un modèle entité-association en modèle relationnel

3. Redondance des données et normalisation

4. Langages de manipulation des données (LMD)

a- Algèbre relationnelle (AR)

b- Calcul relationnel de tuples (CRT)

3.4.a- Algèbre relationnelle (A.R.)

A.R. : Langage de manipulation de données relationnelles

E. CODD (1970)

A.R. : huit opérateurs s'appliquant à une ou deux relations et donnant une relation comme résultat

- Union, intersection, différence
- Restriction (sélection)
- Projection
- Produit cartésien
- Jointure
- Division

Opérateur d'union

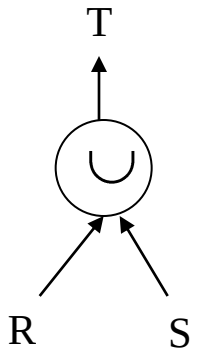
Définition

L'**union** de deux relations R et S de même schéma est une relation T de même schéma contenant l'ensemble des tuples de R et S.

Notation

$T = R \cup S$

T = UNION (R,S)



Exemple d'union de relations

VINS-1

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
110	Mecurey	1978	13
120	Mâcon	1977	12

VINS-2

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
200	Sancerre	1979	11

VINS-3 = UNION (VINS-1,VINS-2)

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
110	Mecurey	1978	13
120	Mâcon	1977	12
200	Sancerre	1979	11

Opérateur d'intersection

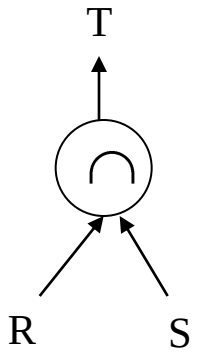
Définition

L'**intersection** de deux relations R et S de même schéma est une relation T de même schéma contenant l'ensemble des tuples appartenant simultanément à R et à S.

Notation

$T = R \cap S$

T = INTERSECT (R,S)



Exemple d'intersection de relations

VINS-1

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
110	Mecurey	1978	13
120	Mâcon	1977	12

VINS-2

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
200	Sancerre	1979	11

VINS-4 = INTERSECT(VINS-1,VINS-2)

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12

Opérateur de différence

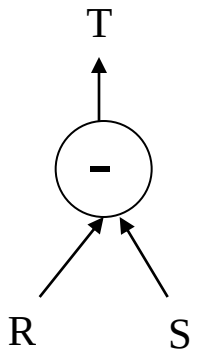
Définition

La **différence** de deux relations R et S de même schéma est une relation T de même schéma contenant l'ensemble des tuples appartenant à R et n'appartenant pas à S.

Notation

T = R - S

T = MINUS (R,S)



Exemple de différence de relations

VINS-1

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
110	Mecurey	1978	13
120	Mâcon	1977	12

VINS-2

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
200	Sancerre	1979	11

VINS-5 = MINUS (VINS-1, VINS-2)

Numéro	Cru	Millésime	Degré
110	Mecurey	1978	13
120	Mâcon	1977	12

Opérateur de projection

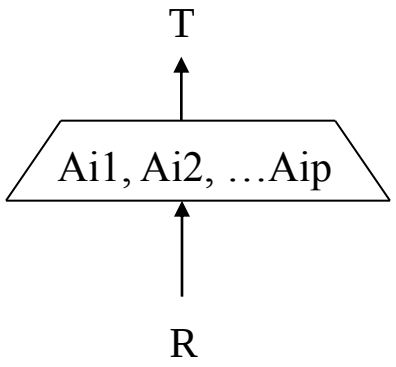
Définition

La **projection** d'une relation R de schéma R(A1, A2,...,An) sur les attributs {Ai1, Ai2, ... , Aip} est une relation R' de schéma R' (Ai1, Ai2, ..., Aip) dont les tuples sont obtenus par élimination des attributs de R n'appartenant pas à R' et par suppression des tuples en double.

Notation

$T = \Pi_{\{Ai1, Ai2, ..., Aip\}} (R)$

T = PROJECT (R / {Ai1, Ai2, ..., Aip})



Exemple de projection de relation

VINS-6

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
110	Mecurey	1978	13
120	Mâcon	1977	12
200	Sancerre	1977	12

VINS-7 = PROJECT (VINS-6 / {Millésime, Degré})

Millésime	Degré
1974	12
1978	13
1977	12

Opérateur de restriction (ou de sélection)

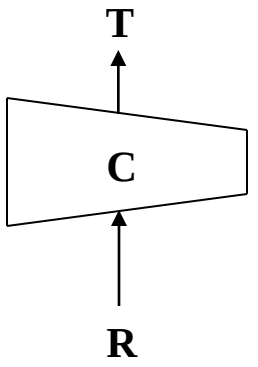
Définition

La restriction d'une relation R à l'aide d'une condition C est une relation R' de même schéma dont les tuples sont ceux de R satisfaisant la condition C.

Notation

$T = \sigma_C (R)$

T = RESTRICT (R / C)



Exemple de restriction de relation

VINS-6

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
110	Mecurey	1978	13
120	Mâcon	1977	12
200	Sancerre	1977	12

VINS-8 = RESTRICT (VINS-6 / Degré = 12)

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
120	Mâcon	1977	12
200	Sancerre	1977	12

Condition de restriction (sélection)

La condition C d'une restriction est une formule logique quelconque avec des connecteurs **ET** (∧) et **OU** (∨) entre conditions simples de la forme $\mathbf{A_i}\ \theta\ \mathbf{a}$

où

- $\mathbf{A_i}$ est un nom d'attribut
- **a** est un élément du domaine de $\mathbf{A_i}$ (constante)
- θ est un des opérateurs **=, <, >, ≠, >=, <=**

ex. `(Cru="Chablis" ∨ Cru="Mâcon")∧ Millésime < 1988`

Produit cartésien

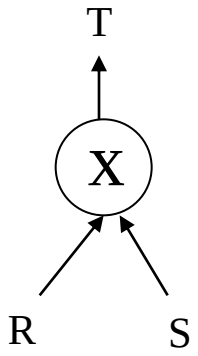
Définition

Le produit cartésien de deux relations R et S (de schémas quelconques) est une relation T ayant pour attributs la concaténation de ceux de R et de S et dont les tuples sont toutes les concaténations d'un tuple de R à un tuple de S (renommage des attributs de même nom).

Notation

T = R x S

T = PRODUCT (R,S)



Exemple de produit cartésien de relations

VINS-2

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
200	Sancerre	1979	11

VITICULTEURS

Nom	Ville	Région
Nicolas	Pouilly	Bourgogne
Martin	Bordeaux	Bordelais

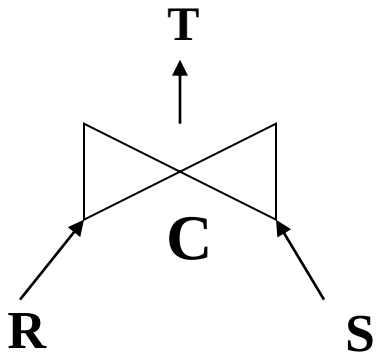
VIGNOBLE = PRODUCT (VINS-2, VITICULTEURS)

Numéro	Cru	Millésime	Degré	Nom	Ville	Région
100	Chablis	1974	12	Nicolas	Pouilly	Bourgogne
100	Chablis	1974	12	Martin	Bordeaux	Bordelais
200	Sancerre	1979	11	Nicolas	Pouilly	Bourgogne
200	Sancerre	1979	11	Martin	Bordeaux	Bordelais

Opérateur de jointure

Définition

La jointure de deux relations R et S selon une condition C est l'ensemble des tuples du produit cartésien R X S satisfaisant la condition C.



Notation

$T = R \downarrow_{C} S$

T = JOIN (R, S/C)

N.B. $R \downarrow_{C} S = \sigma_{C} (R \times S)$

Exemple de jointure de relations

VINS-1

Numéro	Cru	Millésime	Degré
100	Chablis	1974	12
110	Mecurey	1978	13
120	Mâcon	1977	12

VITICULTEURS-BIS

Nom	Ville	Région
Nicolas	Pouilly	Bourgogne
Félix	Mâcon	Bourgogne

V-BIS = JOIN (VINS-1, VITICULTEURS-BIS / Cru = Ville)

Numéro	Cru	Millésime	Degré	Nom	Ville	Région
120	Mâcon	1977	12	Félix	Mâcon	Bourgogne

Jointure naturelle

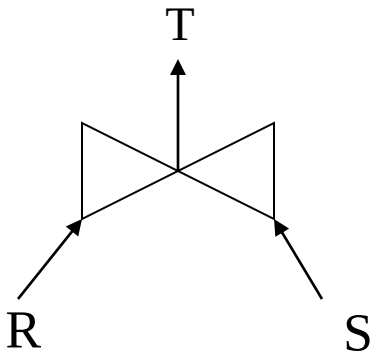
Définition

La jointure naturelle de deux relations R et S est l 'équi-jointure (opérateur d'égalité) de R et S sur tous leurs attributs communs.

Notation

$T = R \downarrow S$

T = JOIN (R, S)



Exemple de jointure naturelle de relations

VINS-1

Numéro	Cru	Millésime	Degré
150	Riesling	1984	11
110	Mecurey	1978	13
120	Mâcon	1977	12

VITIC

Nom	Numéro	Région
Nicolas	150	Alsace
Félix	120	Bourgogne

VINS-C = JOIN (VINS-1, VITIC)

Numéro	Cru	Millésime	Degré	Nom	Région
150	Riesling	1984	11	Nicolas	Alsace
120	Mâcon	1977	12	Félix	Bourgogne

Jointure externe

Jointure externe : inclure les tuples « célibataires » dans la jointure naturelle

- Jointure externe à gauche

  $R = \downarrow S$ (jointure de R et S + «célibataires» de R)

- Jointure externe à droite

  $R \downarrow = S$ (jointure de R et S + «célibataires» de S)

- Jointure externe pleine

  $R = \downarrow = S$ (jointure + «célibataires» de R et de S)

Exemple de jointure externe à gauche :

chercher les informations sur les vins et leur viticulteur en incluant les vins qui n'ont pas de viticulteur

VINS-1

Numéro	**Cru**	**Millésime**	**Degré**
150	Riesling	1984	11
110	Mecurey	1978	13
120	Mâcon	1977	12

VITIC

Nom	**Numéro**	**Région**
Nicolas	150	Alsace
Félix	120	Bourgogne

VINS-C = VINS-1 =⬇ VITIC

Numéro	**Cru**	**Millésime**	**Degré**	**Nom**	**Région**
150	Riesling	1984	11	Nicolas	Alsace
110	**Mecurey**	**1978**	**13**	**null**	**null**
120	Mâcon	1977	12	Félix	Bourgogne

Opérateur de division

Définition

La division d'une relation R(X,Y) par une relation S(Y) est la projection de R sur X restreinte aux tuples en liaison avec **tous** les tuples de S.

Notation

T= R S

Définition formelle

$R(X,Y) \quad S(Y) = T(X) = \{ \langle x \rangle \mid \forall\ y, \langle y \rangle \in S \Rightarrow \langle x,y \rangle \in R \}$

Exemple de division de relations

Produit

numProd
P1
P2
P3

Stock

numProd	numDep
P1	D1
P1	D2
P1	D4
P2	D1
P2	D2
P2	D4
P3	D1
P3	D4

Numéro des dépôts stockant tous les produits :

Stock ÷ Produit

numDep
D1
D4

Exemple de division de relations

Produit

numProd	libellé	pu
P1	K7	5.5
P2	Vis	0.3
P3	Ecrou	0.4

Stock

numProd	numDep	qté
P1	D1	1000
P1	D2	-100
P1	D4	1200
P2	D1	-400
P2	D2	2000
P2	D4	1500
P3	D1	3000
P3	D4	2000

Numéro des dépôts stockant tous les produits :

$$(\Pi_{\{numProd,numDep\}} Stock) \div (\Pi_{\{numProd\}} Produit)$$

numDep
D1
D4

Exemple de requête algébrique (1/2)

Client (numCli, nom, prénom, ddn, rue, CP, ville)
Produit (numProd, libellé, pu, ***numFour***)
Fournisseur (NumFour, raisonSoc)
Commande (***numCli***, ***numProd***, date, quantité)

Ex. de question complexe : Donner les produits commandés en quantité supérieure à 100 et dont le prix dépasse 1000€ . On affichera les numéros de produit, leur libellé et leur prix unitaire ainsi que la date de la commande.

Une requête algébrique = composition d'opérateurs algébriques

Notons **Res** la relation résultat (R_1, R_2, R_3 : relations intermédiaires)

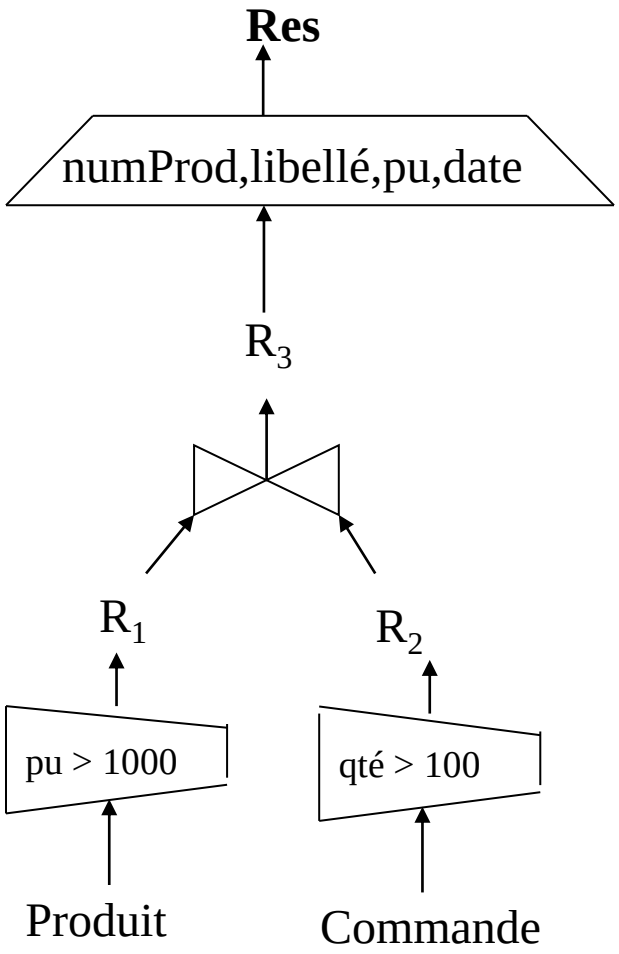
Exemple de requête algébrique (2/2)

$R_1 = \sigma_{pu > 1000}$ (Produit)

$R_2 = \sigma_{qté > 100}$ (Commande)

$R_3 = R1 \downarrow R2$

Res $= \Pi_{\{numProd, libellé, pu, date\}}(R3)$



A vous : construire les requêtes algébriques répondant aux questions (BD VPC)

1) Libellé et prix unitaire de tous les produits
2) Libellé des produits de prix inférieur à 50€
3) Nom et prénom des clients ayant commandé le produits numéro 56.
4) Nom des clients ayant commandé au moins un produit de prix supérieur à 500€.
5) Nom des clients n'ayant pas commandé le produit numéro 56.

A.R. : Ensemble minimal d’opérateurs

• Les requêtes (SQL) dans les SGBDR sont transformées en expressions algébriques

• Cinq opérateurs sont nécessaires (ensemble minimal)

union, différence, projection, produit cartésien, sélection

• Les autres opérateurs s'expriment en fonction des précédents.

A.R. : Quelques propriétés des opérateurs

Quelques règles d'équivalence

- $\sigma_{c1 \wedge c2}(R) = \sigma_{c1}(\sigma_{c2}(R))$
- $\sigma_{c1}(\sigma_{c2}(R)) = \sigma_{c2}(\sigma_{c1}(R))$
- $\Pi_{liste1}(\Pi_{liste2}(R)) = \Pi_{liste1}(R)$ si $liste1 \subseteq liste2$
- $R \downarrow_{C} S = S \downarrow_{C} R$
- $R \downarrow_{C} (S \downarrow_{C'} T) = (S \downarrow_{C} R) \downarrow_{C'} T$

Propriétés utilisées pour l'**optimisation** de requêtes dans les SGBDR

3.4.b Calcul Relationnel de Tuples (CRT)

- CRT = Langage du 1er ordre (CP1) pour manipuler des données relationnelles ayant le même pouvoir d'expression que l'algèbre relationnelle.

- Exemple de requête : $\{ t.a_1, t.a_2 \ldots, t.a_n \mid p(t) \}$
  - Le **résultat** de la requête inclut tous les tuples **t** qui rendent la formule **p(t) vraie**.
  - La **formule** est définie récursivement en partant de **formules atomiques** et en construisant des formules de plus en plus complexes au moyen des **opérateurs (connecteurs) logiques**.

Syntaxe d'un langage du 1er ordre

- L = (A,F)
  - A : Alphabet
  - F : Formules (bien formées) construites sur A
- A contient
  - {Constantes} : a, b, c ...
  - {Variables} : x, y, z ...
  - {Fonctions} avec arité : f( ), g( ), h( ) ....
  - {Prédicats*}avec arité : P( ), Q( ), R( ) ....

*: appelés aussi relations (fonctions booléennes à 2 valeurs Vrai/Faux)

Syntaxe d'un langage du 1er ordre

- Définition d'un terme
  - Toute constante est un terme
  - Toute variable est un terme
  - Si **t1, t2, ..tn** sont des termes et si **f** est une fonction n-aire, alors **f(t1, t2, ..,tn)** est un terme
- Définition d'un atome
  - Si **t1, t2, ..., tn** sont des termes et si **P** est un prédicat n-aire alors **P(t1, t2, .., tn)** est un atome
- Définition d'une formule
  - Tout atome est une formule
  - Si **P** et **Q** sont deux formules alors

    $\neg P$, $P \vee Q$, $P \wedge Q$, $P \Rightarrow Q$, $P \Leftrightarrow Q$, $\forall x\, P(x)$, $\exists x\, P(x)$

    sont des formules

Sémantique d'un langage du 1er ordre

- Exemples de formules (Syntaxe)
  - Grand-père (Jean , Marie)
  - Egal(double(4) , 8)
  - $\forall x$ NOMBRE(x) $\Rightarrow$ ($\exists y$ PLUSGRANDQUE(y, x)
  - $\forall x, \exists y$ (P(x) $\wedge$ Q(y)) $\Rightarrow$ (R(a) $\vee$ Q(b))

- Sémantique : sens d'une formule (V/F)
  - Théorie de la preuve
  - Théorie du modèle : Tables de vérité
    - Quand on attribue des valeurs a chaque terme et à chaque symbole de prédicat dans une formule on dit qu'une interprétation est donnée à la formule ou qu'on l'évalue
      - Résultat : Vrai ou Faux

CRT et langage du 1er ordre

- CRT est un langage du 1er ordre en considérant que dans les formules :
  - Les prédicats comportent :
    - Comparateurs logiques (=, ¬ =, >, >=…)
    - Prédicat unaire pour chaque relation (même nom)
  - Les variables sont associées à des tuples de relations
  - Les termes sont :
    - Constantes associées aux éléments des domaines
    - Fonctions de projection d'une variable de relation sur un de ses attributs (p.nom, p.pu…)

Résultat de la requête **{ $t.a_1$,..., $t.a_n$, $s.b_1$, ...$s.b_k$ | p(t,s) }**

- projections sur les attributs **a_1... a_n** des tuples **t** qui rendent vraie la formule **p(t,s)**
- projections sur les attributs **b_1... b_n** des tuples **s** qui rendent vraie la formule **p(t,s)**

CRT : Variables liées / variables libres

$\forall x (P(x) => Q(x, y))$

x est une variable **liée** et **y** est une variable **libre**.

Une formule ne peut être évaluée que lorsque toutes les variables sont liées.

- Une restriction s'impose sur la définition d'une requête en CRT $\{t.a_1, t.a_2 \ldots, t.a_n \mid p(t)\}$
  - La variable **t** qui apparaît à la gauche de `|' doit être la **seule** variable **libre** dans la formule p(t)

Exemples de requêtes dans le CRT

Soit la BDR de schéma :

Produit (numProd, libellé, pu)

Dépôt (numDep, capacité, adresse)

Stock (numProd, numDep, qté)

La formule **Produit (p)** signifie **p** est un tuple de la relation Produit

La constante **p.numProd** désigne le numéro du produit **p**

- Nom et prix unitaire de tous les produits

{ p.libellé, p.pu | Produit (p) }

- Numéro des produits stockés dans le dépôt 'D2'

{ p.numProd | Produit (p) ∧ ∃s

(Stock(s) ∧ s.numProd=p.numProd ∧ s.numDep = 'D2') }

- Adresse et numéro des dépôts stockant tous les produits

{ d.adr, d.numDep | Dépôt (d) ∧ ∀p

(Produit(p) ⇒ ∃s (Stock(s) ∧ p.numProd=s.numProd ∧ s.numDep=d.numDep))}

Calcul Relationnel de Tuples : Bilan

- Le Calcul relationnel de tuples est non-opérationnel (comme l'Algèbre)

- On exprime dans les requêtes ce qu'on veut obtenir et non comment l'obtenir
  - C'est un langage **déclaratif**

- L'algèbre et le calcul relationnel de tuples ont le même pouvoir expressif (notion de complétude relationnelle)
  - Chaque requête exprimable en algèbre relationnelle l'est aussi en CRT et vice-versa

- SQL dérive de l'algèbre relationnelle et du CRT