Bases de Données Relationnelles

L’algèbre relationnelle

Langages de manipulation

- Langages formels : base théorique solide
- Langages utilisateurs : version plus ergonomique

- Langages procéduraux : définissent comment dériver le résultat souhaité
- Langages assertionnels (ou déclaratifs) : définissent le résultat souhaité

LMD classiques

- Langages formels
  - langages algèbriques : définissent un ensemble d'opérateurs de manipulation
  - langages prédicatifs (calcul) : définissent le résultat souhaité en utilisant des expressions de logique
- Langages utilisateurs
  - inspirés principalement des langages algèbriques : SQL
  - inspirés des langage prédicatifs : QBE, QUEL

L'approche algèbrique

- Une algèbre est un ensemble d'opérateurs de base, formellement définis, qui peuvent être combinés à souhait pour construire des expressions algèbriques
- Propriété des algèbres : fermeture

  Le résultat de tout opérateur est du même type que les opérandes (ce qui est indispensable pour construire des expressions)
- Propriété souhaitée : complétude

  Toute manipulation pouvant être souhaitée par les utilisateurs devrait pouvoir être exprimable par une expression algèbrique

L’algèbre relationnelle

- Opérandes : relations du modèle relationnel (1NF)
- Fermeture : le résultat de toute opération est une nouvelle relation
- Complétude : permet toute opération sauf les fermetures transitives et les fonctions d'agrégation (min, max, count...)
- Opérations unaires (un seul opérande) : sélection (σ), projection (π), renommage (a)
- Opérations binaires (deux opérandes): produit cartésien (´), jointures (*), union (È), intersection (Ç), différence (-), division (/)

Préambule

- Pour chacune de ces 9 opérations, on donne :
  - l'opération
  - la syntaxe (notation)
  - la sémantique (résultat attendu)
  - le schéma
  - d'éventuelles remarques
  - un exemple

Sélection σ

- But : ne retenir que certains tuples dans une relation

Pays	nom	capitale	population	surface
	Autriche	Vienne	8	83
	UK	Londres	56	244
	Suisse	Berne	7	41

On ne veut que les pays dont la surface est inférieure à 100 :

Petit-pays = s [surface < 100] Pays

Petit-pays	nom	capitale	population	surface
	Autriche	Vienne	8	83
	Suisse	Berne	7	41

Sélection s

- Petit-pays = σ [surface < 100] Pays
- Syntaxe : σ [c] R
  c : condition de sélection
- condition-élémentaire :
  attribut opérateur-de-comparaison constante-ou-attribut
  - attribut est un attribut de la relation R
  - opérateur-de-comparaison : =, ? , <, >, =, =
- condition :
  - condition-élémentaire
  - condition ET/OU condition ET Ù OU Ú
  - NON condition NON Ø
  - (condition)

Condition de sélection - Exemples

- σ [nom=capitale] Pays
- Pays dont le nom est le même que celui de sa capitale

- σ [(surface>100 $\wedge$ surface<500) $\vee$ (population>30 $\wedge$ population<300)] Pays
- Pays dont la surface est comprise entre 100 et 500 ou dont la population est comprise entre 30 et 300

Sélection σ

- sémantique : crée une nouvelle relation de population l'ensemble des tuples de R qui satisfont la condition
- schéma (résultat) = schéma (opérande)
- population (résultat) $\subseteq$ population (opérande)

Projection π

- But : ne retenir que certains attributs dans une relation

Pays	**nom**	**capitale**	**population**	**surface**
	Autriche	Vienne	8	83
	UK	Londres	56	244
	Suisse	Berne	7	41

On ne veut que les attributs nom et capitale :

Capitales = p [nom, capitale] Pays

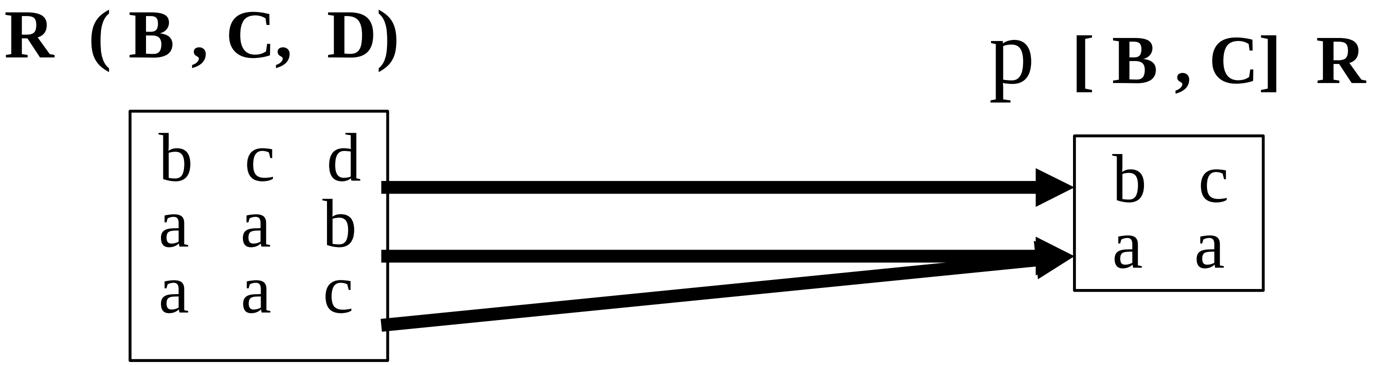
Capitales	**nom**	**capitale**
	Autriche	Vienne
	UK	Londres
	Suisse	Berne

Projection π

- opération unaire
- syntaxe : p **[attributs] R**
  - attributs : liste des attributs de R à conserver dans le résultat
- sémantique : crée une nouvelle relation de population l'ensemble des tuples de R réduits aux seuls attributs de la liste spécifiée
- schéma (résultat) $\subseteq$ schéma (opérande)
- nb tuples (résultat) = nb tuples (opérande)

Effet de bord de la projection

- Création et élimination de tuples en double
  - Une projection qui ne conserve aucun identifiant de la relation peut générer dans le résultat des tuples identiques (à partir de tuples différents de l'opérande)
  - le résultat ne gardera que les tuples différents (fermeture)



trois tuples

deux tuples

Expressions

- On veut les capitales des petits pays:
  - **Petit-pays = s [surface < 100] Pays**
  - **Capitale-petit-pays = p [nom, capitale] Petit-pays**

Capitale-petit-pays =

p [nom, capitale] s [surface < 100] Pays

<u>nom</u>	capitale	population	surface
Irlande	Dublin	3	70
Autriche	Vienne	8	83
UK	Londres	56	244
Suisse	Berne	7	41

(Parties grise et beige à enlever)

Expressions - exemples

Surface-petit-pays =

p **[nom, surface]** s **[surface < 100] Pays**

OU s **[surface < 100]** p **[nom, surface] Pays**

Capitale-petit-pays =

p **[nom, capitale]** s **[surface < 100] Pays**

MAIS s **[surface < 100]** p **[nom, capitale] Pays** **ERREUR !**

π **[nom, capitale]** s **[surface < 100]**
p **[nom, capitale, surface] Pays** **OK**

Renommage a

- but : résoudre des problèmes de compatibilité entre noms d’attributs de deux relations opérandes d’une opération binaire
- opération unaire
- syntaxe : α [nom_attribut1 -> nouveau_nom1, ...] R
- sémantique : les tuples de R avec un (des) nouveau nom d'attribut
- schéma de α [n1–>m1, ..., ni–>mi] R : le même schéma que R avec les attributs n1, ... ni renommés en m1, ... mi
- précondition : les nouveaux noms n’existent pas déjà dans R
- exemple : R2 = α [B–>C] R1

R1

A	B
a	b
y	z
b	b

R2

A	**C**
a	b
y	z
b	b

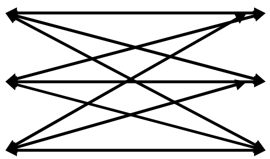
Produit cartésien ´

- but : construire toutes les combinaisons de tuples de deux relations (en général, en vue d'une sélection)
- syntaxe : R ´ S
- exemple :

R

A	B
a	b
b	c
c	b

n tuples



S

C	D	E
c	d	e
b	a	b
a	a	c

m tuples

R ´ S

A	B	C	D	E
a	b	c	d	e
a	b	b	a	b
a	b	a	a	c
b	c	c	d	e
b	c	b	a	b
b	c	a	a	c
c	b	c	d	e
c	b	b	a	b
c	b	a	a	c

n x m tuples

Produit cartésien

- opération binaire
- sémantique : chaque tuple de R est combiné avec chaque tuple de S
- schéma : schéma $(R \times S)$ = schéma(R) $\cup$ schéma(S)
- précondition : R et S n'ont pas d'attributs de même nom
  (sinon, renommage des attributs avant de faire le produit)

Jointure naturelle $\bowtie$ ou *

- but : créer toutes les combinaisons significatives entre tuples de deux relations
  - ◆ significatif = ont la même valeur pour **tous** les attributs de même nom
- précondition : les deux relations ont au moins un attribut de même nom
- exemple :

R

A	B
a	b
b	c
c	b

S

B	C	D
b	c	d
a	a	b
d	a	c

R * S

A	B	C	D
a	b	c	d
c	b	c	d

Jointure naturelle $\bowtie$ ou *

- opération binaire
- syntaxe : R * S
- sémantique : combine certains tuples
- schéma : schéma (R * S) = schéma (R) $\cup$ schéma (S)
  - les attributs de même nom n'apparaissent qu'une seule fois
- la combinaison exige l'égalité des valeurs de tous les attributs de même nom de R et de S
  - si R et S n'ont pas d'attributs de même nom la jointure peut être dynamiquement remplacée par un produit cartésien

Jointure naturelle - Exemple

- Pays (<u>nom</u>, capitale, population, surface, continent)

  LangueParlée (<u>langue, pays</u>, %population)

- Pour chaque langue, dans quels continents est-elle parlée ?

- π [langue, continent]
  (LangueParlée * α [nom –> pays] Pays)

Theta jointure $\bowtie$[c] ou *[c]

- but : créer toutes les combinaisons significatives entre tuples de deux relations
  - significatif = selon un critère de combinaison explicitement défini en paramètre de l’opération (c)
- précondition : les deux relations n’ont pas d’attribut de même nom
- exemple : R * [B ? C] S

R

A	B
a	b
b	c
c	b

S

C	D	E
b	c	d
b	a	b
c	a	c

R * [B ? C] S

A	B	C	D	E
a	b	c	a	c
b	c	b	c	d
b	c	b	a	b
c	b	c	a	c

Theta-jointure

- opération binaire
- syntaxe : R *[c] S
  - c : condition de jointure
- condition-élémentaire :

  attribut1   opérateur-de-comparaison   attribut2
  - attribut1 est un attribut de la relation R
  - attribut2 est un attribut de la relation S (ou vice-versa)
  - opérateur-de-comparaison : =, ? , <, >, =, =
- condition :
  - condition-élémentaire
  - condition ET/OU condition   ET Ù   OU Ú
  - NON condition   NON Ø
  - (condition)

Theta-jointure (suite)

- sémantique : combine les tuples qui satisfont la condition
- schéma (R *[c] S) = schéma (R) $\cup$ schéma (S)
- Exemple :

  Pays (<u>nom</u>, capitale, population, surface, continent)

  LangueParlée (<u>langue, pays</u>, %population)

Pour chaque langue, dans quels continents est-elle parlée ?

π [langue, continent]
(LangueParlée *[nom = pays] Pays)

Jointures : opérateurs dérivés

- R (A, B, C)
- S (C, D, E)
- R*S = π [A, B, C, D, E] σ [CC=C] ((α [C–>CC] R) x S)
- Même chose pour la theta-jointure

Union È

- opération binaire
- syntaxe : R ∪ S
- sémantique : réunit dans une même relation les tuples de R et ceux de S
- schéma(R ∪ S) = schéma(R) = schéma(S)
- précondition : schéma(R) = schéma(S)
- Exemple :

R1

A	B
a	b
b	b
y	z

R2

A	B
u	v
y	z

R1 È R2

A	B
a	b
b	b
y	z
u	v

Union (suite)

- Effet de bord : des tuples en double peuvent être créés.
- Ils sont automatiquement supprimés du résultat.

Intersection Ç

- opération binaire
- syntaxe : $R \cap S$
- sémantique : sélectionne les tuples qui sont à la fois dans R et S
- schéma $(R \cap S)$ = schéma (R) = schéma (S)
- précondition : schéma (R) = schéma (S)
- Exemple :

R1

A	B
a	b
y	z
b	b

R2

A	B
u	v
y	z

R1 Ç **R2**

A	B
y	z

Différence -

- opération binaire
- syntaxe : R – S
- sémantique : sélectionne les tuples de R qui ne sont pas dans S
- schéma (R – S) = schéma (R) = schéma (S)
- précondition : schéma (R) = schéma (S)
- Exemple :

R1

A	B
a	b
y	z
b	b

R2

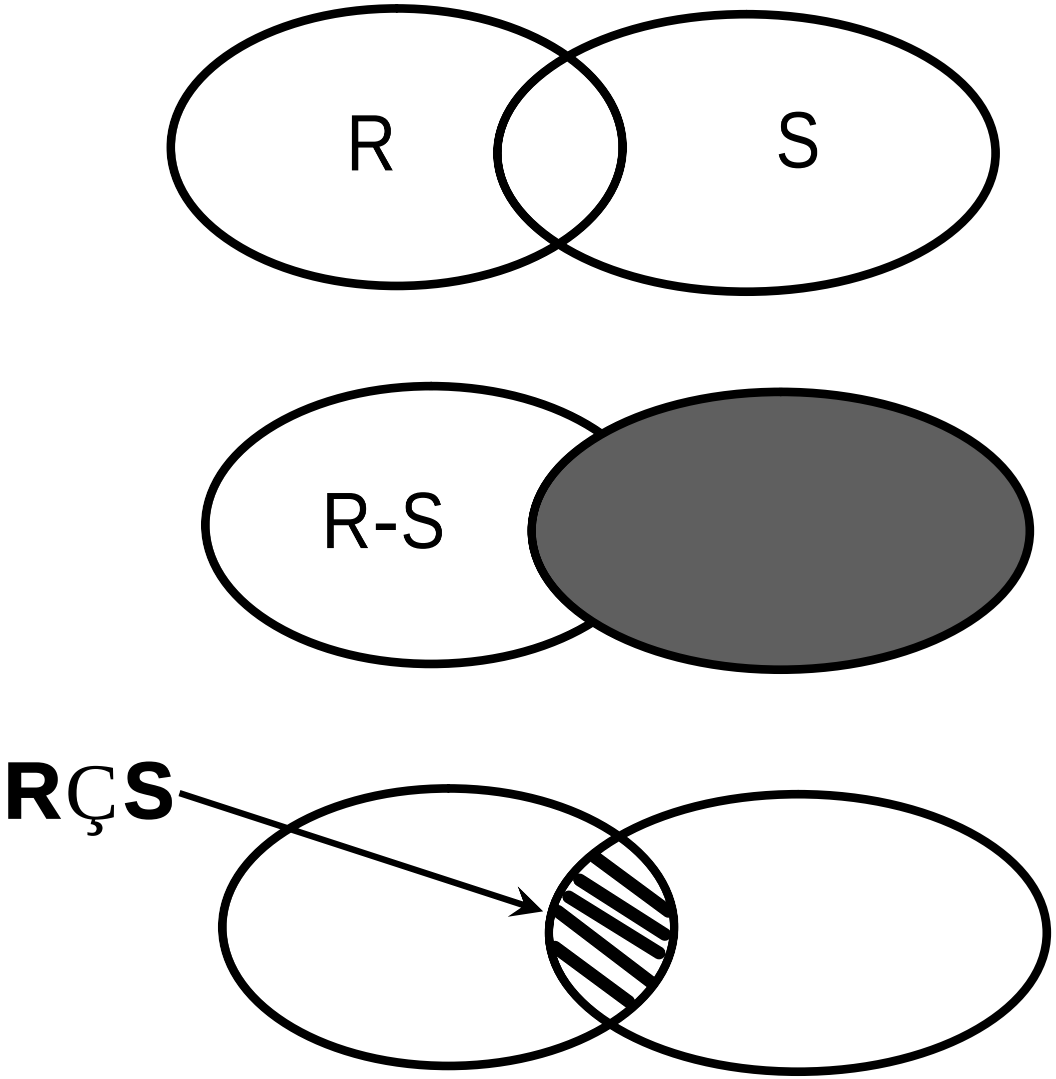
A	B
u	v
y	z

R1 – R2

A	B
a	b
b	b

Intersection : opérateur dérivé

- $R \cap S = R - (R - S)$



populations

La division /

- But : traiter les requêtes du genre «les ... tels que TOUS les ...»
- Soient R(A1, ..., An) et V(A1, ..., Am)
  avec n>m et A1, ..., Am des attributs de même nom dans R et V
- R/V = les tuples de R
  réduits aux attributs Am+1, ..., An
  tels qu'ils existent dans R concaténés à **tous** les
  tuples de V
- $R/V = \{ <am+1, am+2, ..., an> /$
  $\forall <a1, a2, ..., am> \in V$
  $\exists <a1, a2, ..., am ,am+1, am+2, ..., an> \in R \}$

La division - exemples

R

A	B	C
1	1	1
1	2	0
1	2	1
1	3	0
2	1	1
2	3	3
3	1	1
3	2	0
3	2	1

V

B	C
1	1
2	0

R / V

A
1
3

V'

B	C
1	1

R / V'

A
1
2
3

V''

B	C
3	5

R / V''

A
/

Exemple de division

R : Obtenu

ETUDIANT	COURS
Francois	BD
François	SI
Francois	Prog
Annie	BD
Annie	SI
Annie	Math
Pierre	Prog
Pierre	BD

V : Prérequis

COURS
Prog
BD

R/V

ETUDIANT
Francois
Pierre

R/V : Etudiants ayant tous les prérequis

Division - opérateur dérivé

- $R(A_1, \ldots, A_n)$ et $V(A_1, \ldots, A_m)$
  avec $n>m$

- $R \,/\, V =$
  $\pi[A_{m+1}, \ldots, A_n)]\ R$
  $-$
  $\pi[A_{m+1}, \ldots, A_n)]\ (\ (\ (\pi[A_1, \ldots, A_m)]R) \times V\ ) - R\ )$

Propriétés des opérateurs

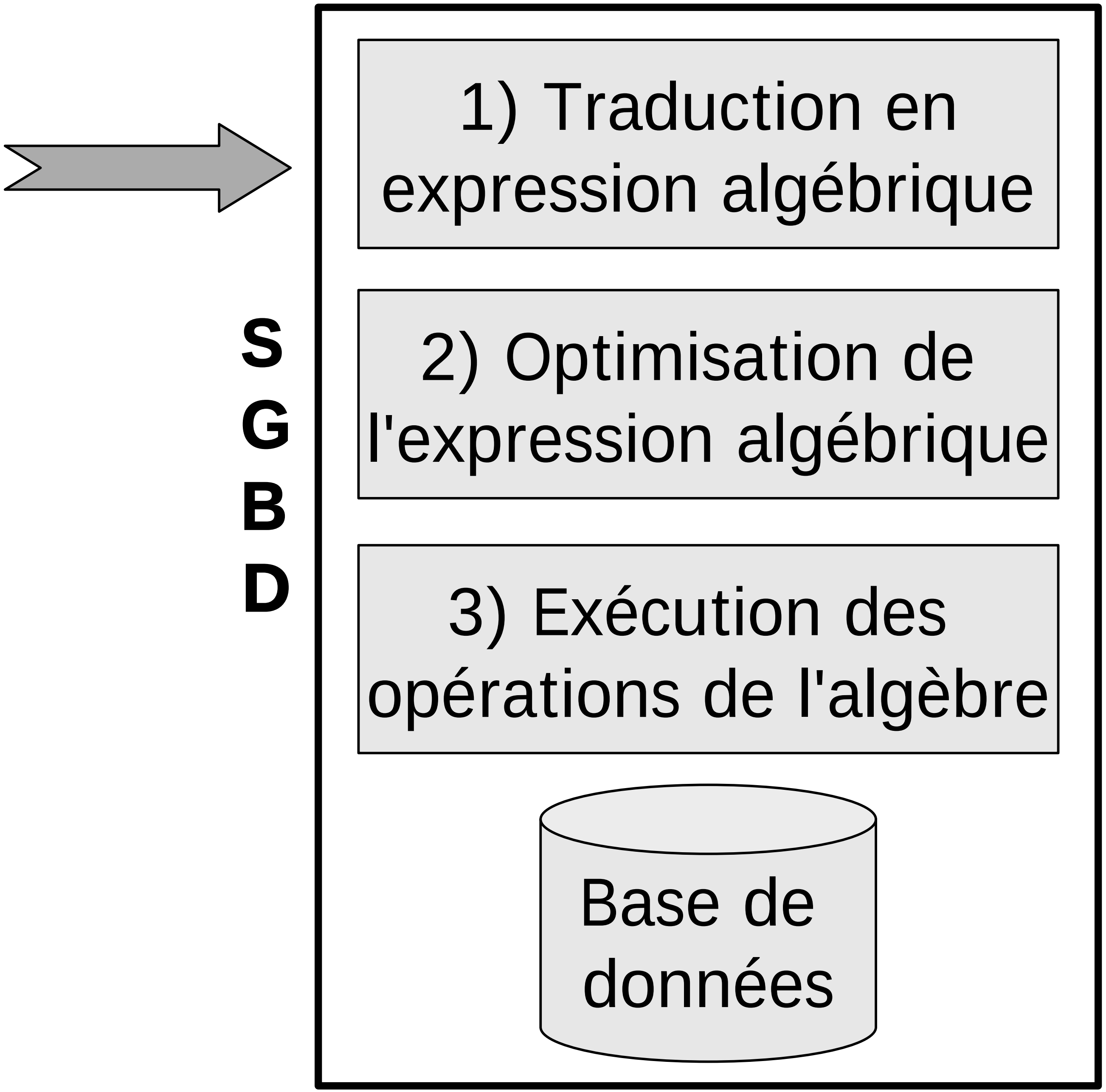
- $R \cap S = S \cap R$ (commutativité)
- $R * S = S * R$
- ...
- σ [p1] (σ [p2] R) = σ [p2] (σ [p1] R) = σ [p2 $\wedge$ p1] R
- σ [p] (π [a] R) = π [a] (σ [p] R) si attributs(p) $\subseteq$ a
- ...

Utilité : optimisation des expressions

- faire les sélections et projections le plus tôt possible pour réduire le volume des relations opérandes

Optimiseur de requêtes

SELECT ...
FROM ...
WHERE ...



Exemples de requêtes algébriques

- Soient les relations suivantes :

Journal (<u>code-j</u>, titre, prix, type, périodicité)

Dépôt (<u>no-dépôt</u>, nom-dépôt, adresse)

Livraison (<u>no-dépôt, code-j, date-liv</u>, quantité-livrée)

Répondre aux requêtes suivantes :

- Quel est le prix des journaux ?

  π [titre, prix] Journal

- Donnez tous les renseignements connus sur les journaux hebdomadaires

  σ [périodicité = "hebdomadaire"] Journal

Répondre aux requêtes suivantes (2)

- Donnez les codes des journaux livrés aux dépots de Bienne

Bienne

Dépôt (<u>no-dépôt</u>, nom-dépôt, adresse)

Livraison (<u>no-dépôt , code-j, date-liv</u>, quantité-livrée)

?

π [code-j] (σ [adresse = "Bienne"] Dépôt * Livraison)

Répondre aux requêtes suivantes (3)

- Donnez les numéros des dépots qui reçoivent plusieurs journaux

Livraison (no-dépôt , code-j, date-liv, quantité-livrée)

= ?

Livraison (no-dépôt , code-j, date-liv, quantité-livrée)

?

π [no-dépôt] (

α [no-dépôt -> dépôt2 , code-j -> code2] Livraison

* [code-j $\neq$ code2 $\wedge$ no-dépôt=dépôt2]

π [no-dépôt, code-j] Livraison)