Modèle et langages relationnels

1. Le modèle de données relationnel

- Relations
- Algèbre relationnelle
- Optimisation

2. Le calcul relationnel

- Langages de requêtes vs. Langages de programmation
- Langages théoriques vs. langages pratiques
- Requêtes simples vs. complexes
- Consultation vs. Mises à jour

3. Le mécanisme de vues relationnelles

- Consultation vs. Mises à jour

Le modèle relationnel (1/2)

La majorité des modèles de données utilisent plusieurs concepts

Entités, Associations (et les Contraintes d'Intégrité)

Le modèle relationnel utilise un seul concept : la relation ***(et, ensuite, les C.I.)***

nom de relation

attribut

VINS	NV	CRU	MILL
	1	Mercurey	1982
	2	Mercurey	1983
	4	Bourgogne	1983

schéma

tuple

domaine

Le modèle relationnel (2/2)

- **Définition syntaxique :**
  - **domaine** (= type nommé)
  - **relation** (= sous-ensemble nommé, variant ou non, du produi cartésien de domaines)
  - schéma d'une relation = relation définie en intention (= liste de ses domaines, aussi appelés attributs))
  - instanciation d'une relation = relation définie en extension (= ensemble des n-uplets d'une instance, aussi appelés "**tuples**")
  - jamais deux tuples identiques (set_of <> bag_of)
- **Définition sémantique objet :**
  - attribut = fonction: objet -> domaine
  - schéma = type complexe (= liste des attributs) d'un objet
  - relation = ensemble de tous les objets de même schéma (= tuples)

Schéma, instance et requêtes relationnelles

VINS	NV	CRU	MILL
	1	Mercurey	1982
	2	Mercurey	1983
	4	Bourgogne	1983

PROD	NV	NVT
	1	10
	2	10
	1	11
	4	10

VITIC.	NVT	NOM	PRENOM	VILLE
	10	Dupont	Michel	Mercurey
	11	Durand	Jean	Beaunes

- Trouver les noms et prénoms de tous les viticulteurs
- Trouver tous les millésimes des Mercurey
- Trouver tous les crus de 1983 et leur producteurs

L'algèbre relationnelle

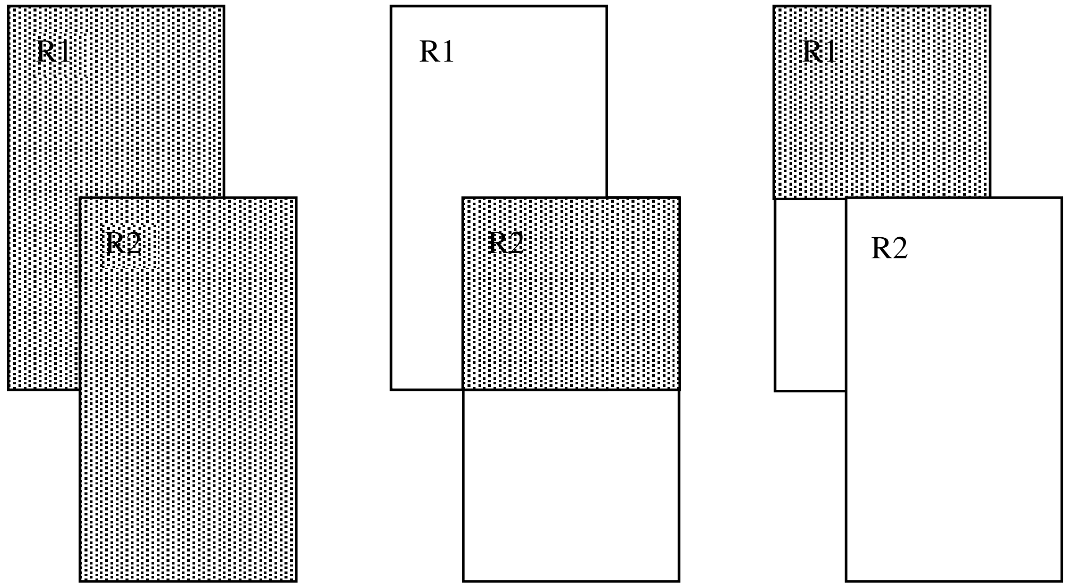
opérations définies sur les relations pour les manipuler :

les opérateurs relationnels

OPERATEURS ENSEMBLISTES	OPERATEURS RELATIONNELS
. UNION	. RESTRICTION
. INTERSECTION	. PROJECTION
. DIFFERENCE	. JOINTURES
. PRODUIT CARTESIEN	. DIVISION

Union, intersection, différence

Ce sont les opérations classiques de la théorie des ensembles. Elles ne s'appliquent qu'à des **relations de même schéma**.



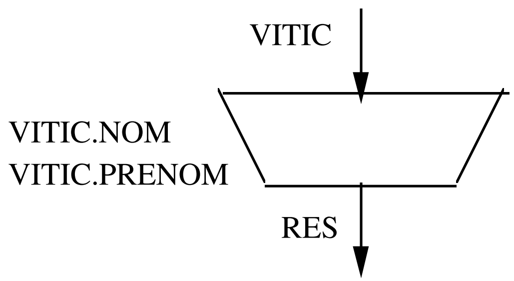
R1 UNION INTERSECTION DIFFERENCE R2

Opérateurs relationnels : la projection

C'est une opération d'extraction de certaines colonnes désignées d'une relation.
La relation créée a pour schéma un sous-schéma de celui de la relation source

N.B. élimination éventuelle des doublons.

VITIC.	NVT	NOM	PRENOM	VILLE
	10	Dupont	Michel	Mercurey
	11	Durand	Jean	Beaunes



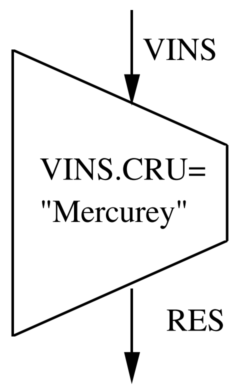
Trouver les Noms et Prénoms de tous les viticulteurs

Opérateurs relationnels : la restriction

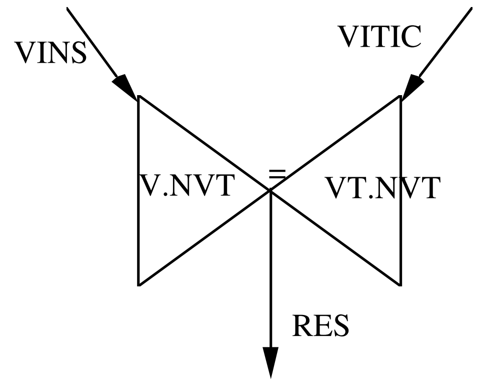
C'est une opération de sélection d'un sous-ensemble de tuples d'une relation selon un ou plusieurs critères (= != < >) exprimés sur les attributs. La relation créée a même schéma que la relation source.

VINS	NV	CRU	MILL	NVT
	1	Mercurey	1982	10
	2	Mercurey	1983	10
	3	Mercurey	1982	11
	4	Bourgogne	1983	10

Trouver tous les vins du cru Mercurey



Opérateurs relationnels : la jointure



C'est une restriction du produit cartésien de deux relations sources. Pour obtenir la relation résultat on opère de la façon suivante:

1) on réalise le produit cartésien de R1 et R2; on obtient une relation R3 ayant pour schéma la réunion des deux schémas des relations sources.

2) on réalise une restriction sur R3 selon un critère comparant une colonne de R1 avec une colonne de R2.

Plusieurs types de jointure :

jointure naturelle :

équi-jointure de deux relations sur deux attributs identiques

théta-jointure :

restriction du produit cartésien de deux relations avec un critère de restriction de la forme :

<attribut1> <opérateur> <attribut2>

auto-jointure :

jointure d'une relation avec elle-même

5 opérateurs de base

1) UNION $R \cup S$

2) DIFFERENCE $R - S$

3) PRODUIT CARTESIEN $R \times S$

4) RESTRICTION $\sigma_E (R)$

5) PROJECTION $\pi_{A1,\ldots, Am} (R)$

OPERATEURS DERIVES

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

2) θ - JOINTURE $R \infty_{(i\ \theta\ j)} S = \pi_{(r+s-j)} (\sigma_{i\ \theta\ j} (R \ X \ S))$

3) DIVISION $R \div S = \pi_{i1,\ldots, i(r-s)} (R) - \pi_{i1,\ldots, i(r-s)} ((\pi_{i1,\ldots, i(r-s)} (R) \ X \ S) - R)$

Exemple : la base Coopérative

SYNONYMES EMPLOYES

VINS	(NV,CRU,MIL)	-> VINS = V
VITICULTEURS	(NVT,NOM,PRENOM,VILLE)	-> VITICULTEURS = VT
PRODUCTIONS	(NV,NVT)	-> PRODUCTIONS = P
BUVEURS	(NB,NOM,PRENOM,VILLE)	-> BUVEURS = B
COMMANDES	(NC,DATE,NV,QTE,NB)	-> COMMANDES = C
EXPEDITIONS	(NC,DATE,QTE)	-> EXPEDITIONS = E

Recherche Mono relation

"DONNER LES NUMEROS DES VINS DE MILLESIME 1975"

1) TEMP <- $\sigma_{\text{MILL=1975}}$ (V)
2) RESU <- $\pi_{\textbf{NV}}$ (TEMP)

ce qui s'écrit encore :

RESU <- $\pi_{\textbf{NV}}$ ($\sigma_{\text{MILL=1975}}$ (V))

Recherche Multi-relations

"DONNER LES NOMS DES VITICULTEURS QUI PRODUISENT DU Muscadet"

1) $T1 \leftarrow \sigma_{\text{CRU='Muscadet'}} (V)$

2) $T2 \leftarrow T1 \infty_{(NV)} P$

3) $T3 \leftarrow T2 \infty_{(NVT)} VT$

4) $R \leftarrow \pi_{NOM} (T3)$

ce qui s'écrit encore :

$R \leftarrow \pi_{NOM} ($

$((\sigma_{\text{CRU='Muscadet'}} (V) \infty_{(NV)} P),$

$\infty_{(NVT)} VT)$

$) ;$

Arbre algébrique

une requête est décrite par une composition d'opérateurs de l'algèbre relationnelle

une description graphique est plus lisible :

utilisation d'arbres algébriques

un arbre algébrique est un arbre représentant une requête avec :

-> noeuds feuilles = relations de base
-> noeuds intermédiaires = opérateurs
-> noeuds racine = résultat
-> arcs = flux de données

Arbre Algébrique d'exécution

"DONNER LES NOMS ET LES PRENOMS DES BUVEURS HABITANT Paris QUI ONT COMMANDE DU Mâcon 1977 AVANT LE 15/05/88"

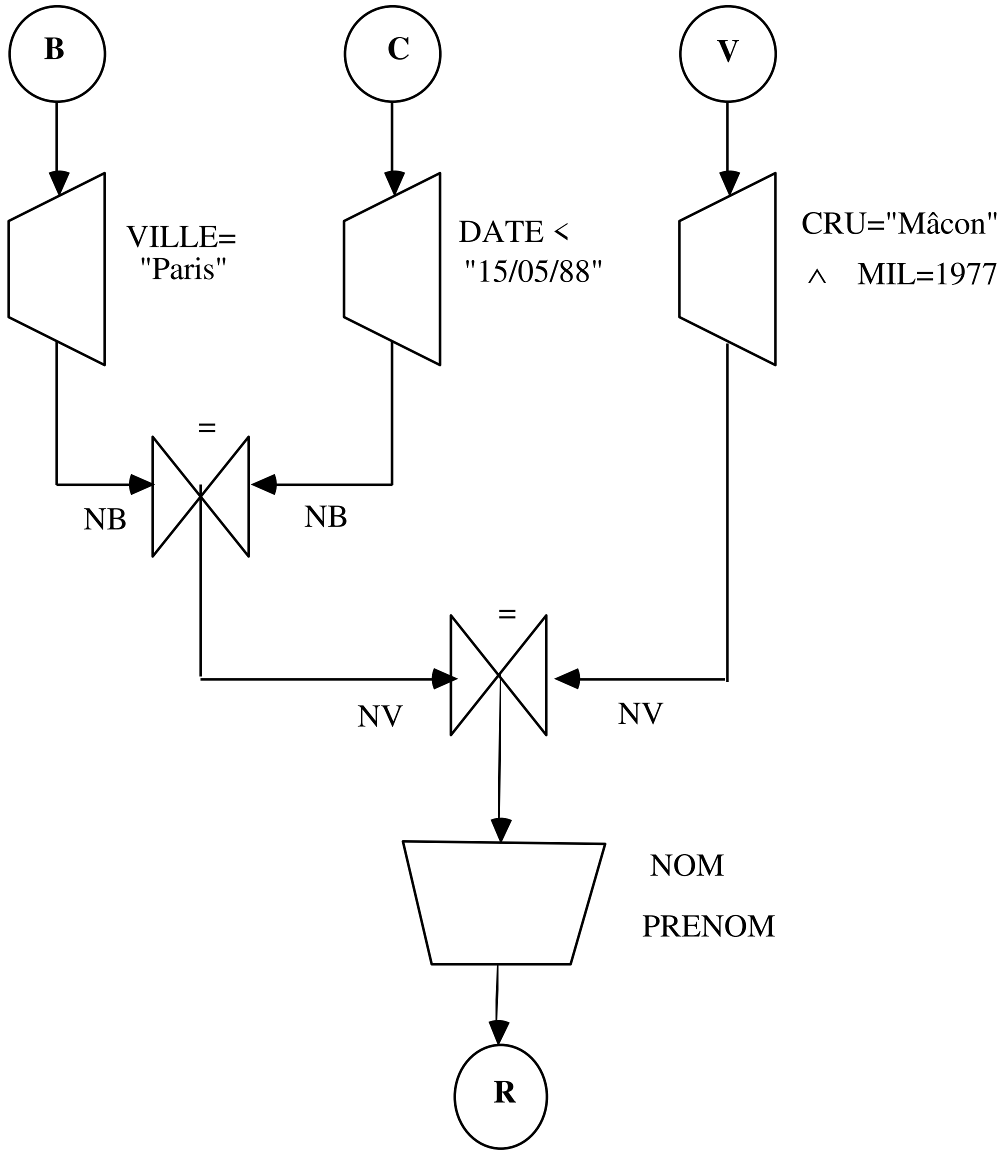
==>

ON PEUT REPONDRE DE DIVERSES FACONS

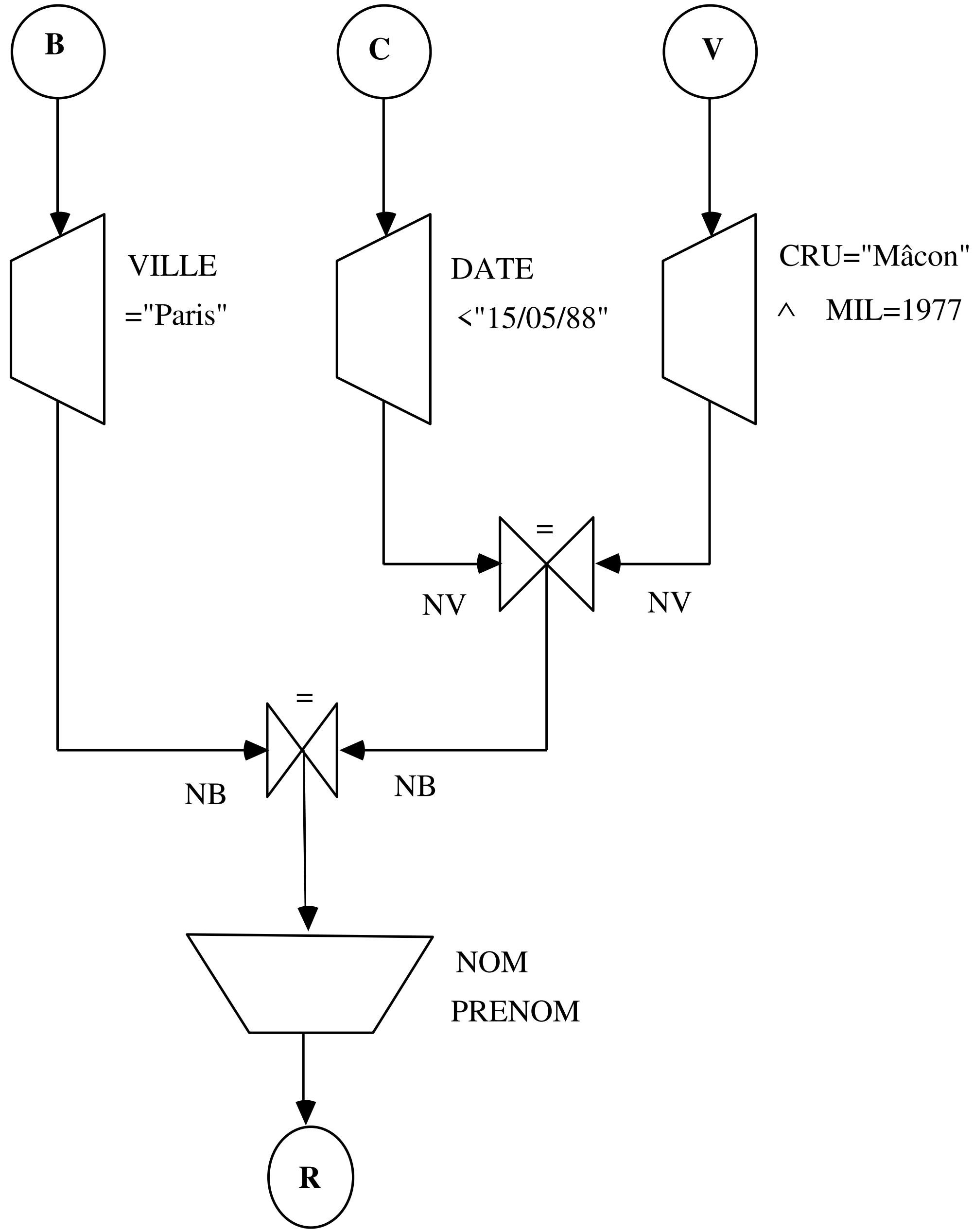
==>

CHOISIR LA MEILLEURE = OPTIMISATION

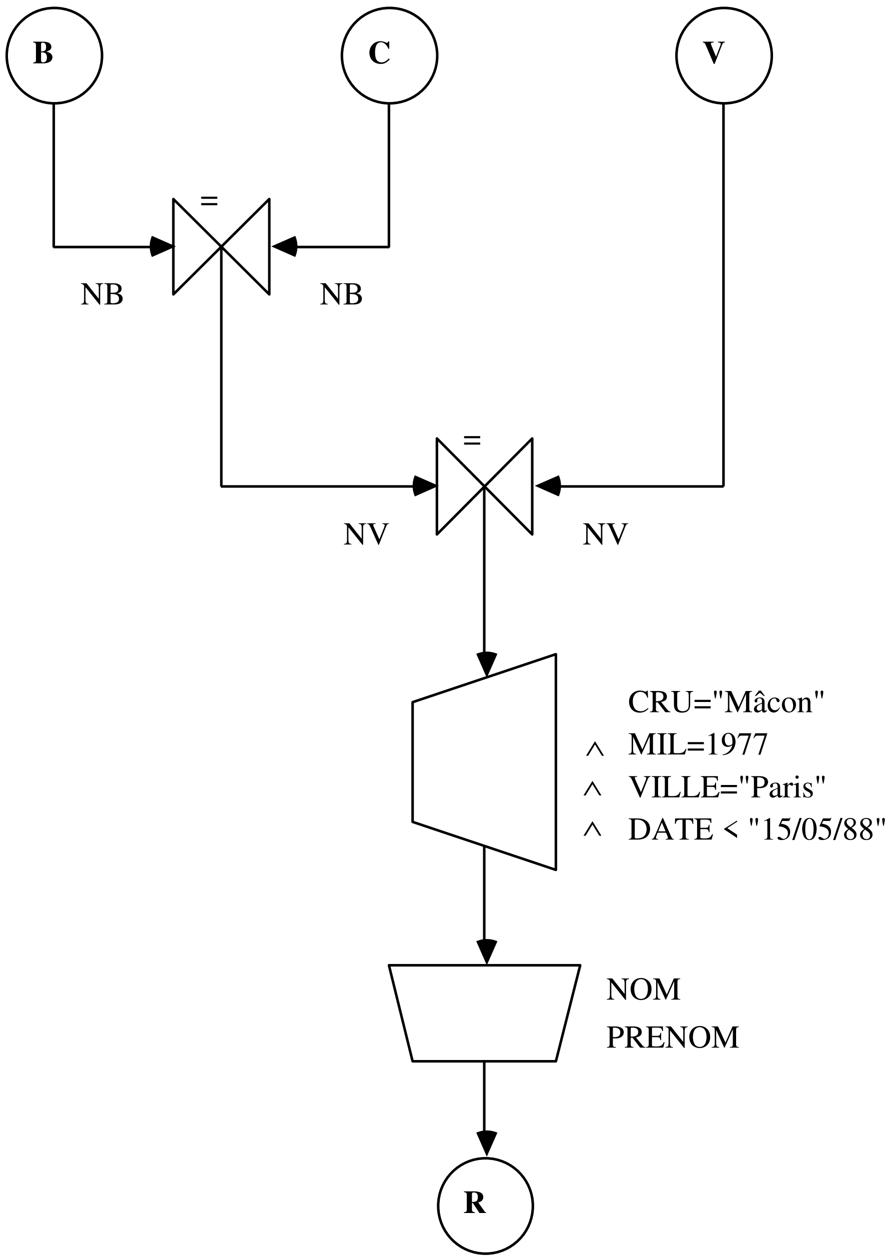
Première solution



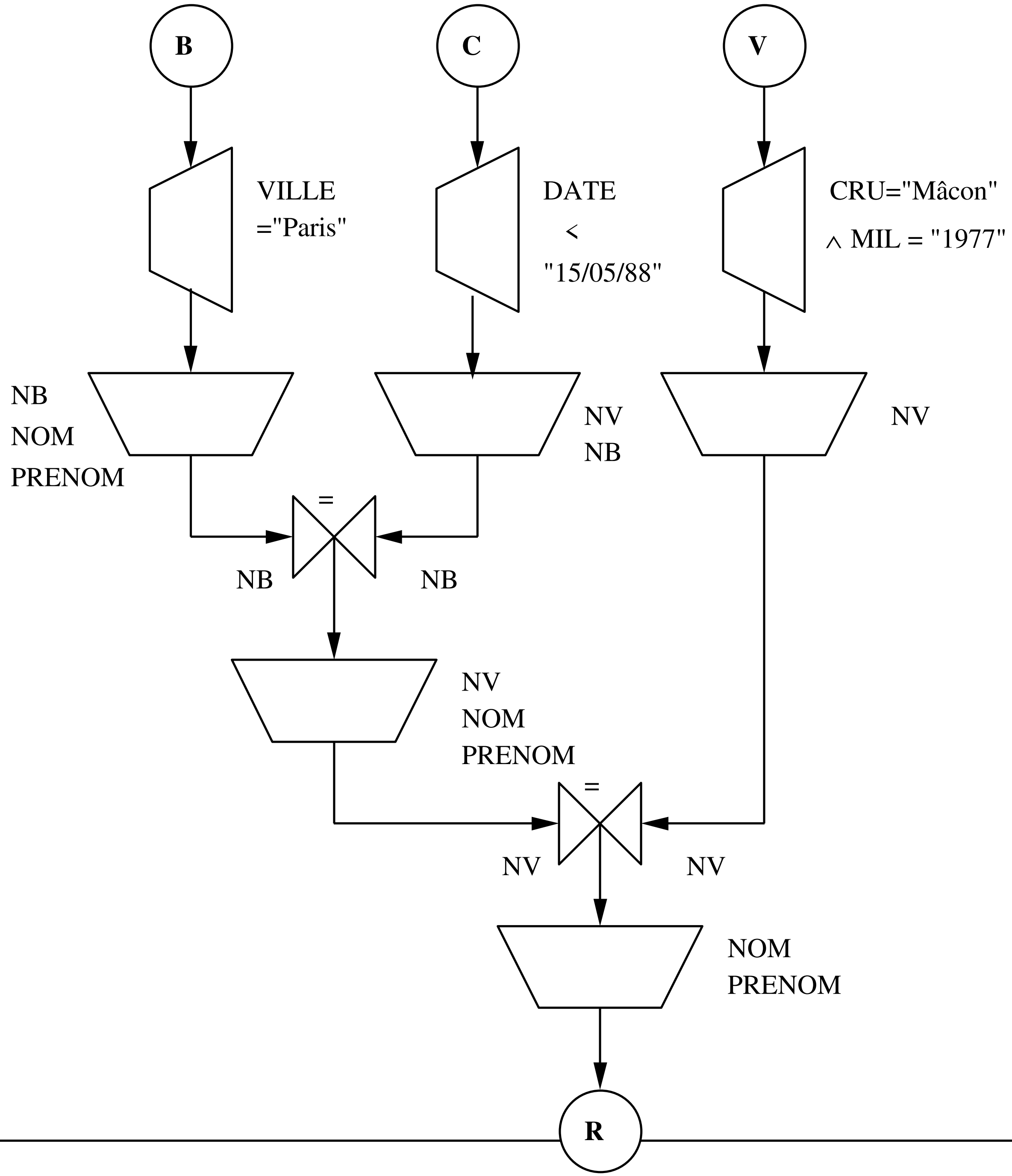
Deuxième solution



Troisième solution



Quatrième solution



Optimisation

• le SGBD doit exécuter un arbre algébrique optimisé

• optimiser

  -> le nombre d'E/S

  -> le temps UC

  -> …

• facteurs déterminants

  -> ordre d'exécution des opérations algébriques

  -> algorithmes implantant les opérations algébriques

  -> placement des données sur disque

  -> taille des relations intermédiaires

Mises à jour en Alg. Rel.

."INSERTION D'UN VIN Jurançon 1979 AVEC LE NUMERO 150"

$$V \leftarrow V \cup \{ (150, \text{"Jurançon"}, 1979) \}$$

."SUPPRESSION DES VINS DE MILLESIME 1980"

$$V \leftarrow V - (\sigma_{MILL=1980} (V))$$

Exercice (1/4)

- R1(Parent, Enfant)
- R2(Personne, Age, Sexe)
- R3(Enfant, Ecole)

domaine(Ecole | Personne) = varchar()

domaine(Parent | Enfant) $\in$ domaine(Personne)

domaine(Sexe) = {'F', 'M'}

domaine(Age) = [0, 150]

N.B.

Ri' désignera ci-après une deuxième instanciation de Ri, Ri'' une troisième.

Exercice (2/4)

- 1) quels sont les enfants de Pierre?

R1(Personne) = $\pi_{\text{Enfant}}(\sigma_{\text{Parent='Pierre'}}(\text{R1}))$

- 2) quels âges ont les enfants de Marie?

R2(Personne, Age) = $\pi_{\text{Enfant, Age}}(\sigma_{\text{Parent='Marie'}}(\text{R1}))$

- 5) quels sont les grands-parents de Jacques?

R5(Personne) = $\pi_{\text{R1'.Parent}}(\text{R1'}\infty_{\text{Enfant=Parent}}(\sigma_{\text{Enfant='Jacques'}}(\text{R1}))$

Exercice (3/4)

- 6) quels parents n'ont que des garçons?

R6(Personne) = π_{Parent}(R1') - π_{Parent}(R1 $\infty_{Enfant=Personne}$ ($\sigma_{Sexe='F'}$(R2)))

- 7) quels couples de parents ont au moins deux enfants?

R(Parent1, Parent2, Enfant) = R1' $\infty_{Enfant=Enfant \wedge R1'.Parent>R1.Parent}$(R1)

R7(P1, P2, E) = R' $\infty_{Parent1=Parent1 \wedge Parent2=Parent2 \wedge R'.Enfant>R.Enfant}$(R)

- 8) liste des parents et des écoles de leurs enfants

R8(Parent, Ecole) = $\pi_{Parent,Ecole}$(R1 $\infty_{Enfant=Enfant}$(R3))

- 9) liste des parents qui ont au moins un enfant dans chacune des écoles

R9(Personne) = R8 $\div$ π_{Ecole}(R3)

Exercice (4/4)

- 10)quels sont les oncles de Jacques?

$R(P) = \pi_{Enfant}(R1 \propto_{Parent=Personne}(R5)) - \pi_{Parent}(\sigma_{Enfant='Jacques'}(R1))$

$R10 = \pi_{Personne}(\sigma_{Sexe='M'}(R2 \propto_{Personne=P}(R\ ))$

- 11)qui n'a pas d'enfant scolarisé?

$R11(Personne) = \pi_{Parent}(R1' - R1 \propto_{Enfant=Enfant}(R3))$

- 12)quels parents ont au moins un fils plus âgé qu'une fille?

$R(Parent, Enfant, Age, Sexe) = R1 \infty_{Enfant=Personne}(R3)$

$R12 = \pi_{Parent}(R' \infty_{Parent=Parent \wedge R'.Sexe>R.Sexe \wedge R'.Age>R.Age}(R)$

- 13)quels sont les arrière-grands-parents de Bérénice?

$R13 = \pi_{Parent}(R1'' \propto_{Enfant=Parent}(R1' \propto_{Enfant=Parent}(\sigma_{Enfant='Bérénice'}(R1))))$

N.B. $\propto$ désigne une semi-jointure

Modèle et langages relationnels

1. Le modèle de données relationnel

- Relations
- Algèbre relationnelle
- Optimisation

2. Le calcul relationnel

- Langages de requêtes vs. Langages de programmation
- Langages théoriques vs. langages pratiques
- Requêtes simples vs. complexes
- Consultation vs. Mises à jour

3. Le mécanisme de vues relationnelles

- Consultation vs. Mises à jour

Problème

Un utilisateur ne peut prendre en compte l'ensemble des paramètres des heuristiques/algorithmes d ’optimisation quand il veut interroger la BD.

-> trouver un langage permettant d'exprimer une question sans indiquer un enchainement d'opérations algébriques

==> langage assertionnel (non procédural)

le SGBD assurera :

1) la traduction sous la forme d'un arbre algébrique

2) l'optimisation de l'arbre algébrique

3) l'exécution de l'arbre algébrique

Les langages relationnels

- 2 types de langages

  Alg. rel. vs. calc. rel.

  comment quoi

-> Intérêt de la théorie : on mesure la puissance d'expression d'un langage

-> Logique du 1° ordre : les deux langages sont équivalents

- Intérêt dans le cadre des BD

  Niveaux d'abstraction d'une Base de Données.

- Ce ne sont pas des langages pour les utilisateurs

  -> Langage interne du SGBD : Alg. Rel.

  -> Langage utilisateurs (administrateur, développeur d'applications, utilisateur final) : SQL, QBE…

L'interrogation avec le modèle relationnel

Basée sur la logique des prédicats du premier ordre

Le calcul relationnel de tuple

Formules composées à partir

de termes atomiques :

variable: t, appartenance: $t \in R$, condition: t(A) = valeur

des quantificateurs:

(quel que soit $\forall$, il existe $\exists$) et des connecteurs (et, ou, non)

formules correctes: $\phi(t)$

Requêtes en calcul relationnel de tuples $\{t : <A1,A2,..,An> / \phi(t) \}$

SQL

Version industrielle et commerciale du calcul relationnel

La logique des prédicats du premier ordre

Les variables sont des tuples

variable: t, appartenance: **t** $\in$ **R**, RANGE OF t IS R , R(t)

Les formules de base

condition $\phi(t)$: **t.A** θ **valeur**, t(A) θ valeur, A(t) θ valeur

condition $\phi(t, u)$: **t.A** θ **u.B**

où θ désigne un opérateur compatible avec les types de **t.A** et **u.B**

Les formules valides avec connecteurs logiques

(ϕ), nonϕ , ϕ ou π

ϕ et π = non (nonϕ ou nonπ)

ϕ implique π = nonϕ ou π

Les variables "libres" peuvent être **"liées"** par des quantificateurs:

quel que soit: $\forall_t \phi(t)$, il existe $\exists_u \pi(u)$

Expression : {t : <u.A1,v.A2,..,z.An> / ϕ(u, v, .., z) }

{schéma(variables libres) du tuple résultat / formule valide(variables libres et liées }

Exemples

calcul relationnel de tuple

<u>Quels sont les viticulteurs qui ont produit au moins un vin en 1983 (nom et cru) ?</u>

{t : cru, nom / $\exists$ v $\in$ vins, $\exists$ p $\in$ prod, $\exists$ w $\in$ vitic, t(cru) = v(cru) and t(nom)=w(nom) and w(nvt) = p(nvt) and p(nv) = v(nv) and v(mill) =1983}

SQL *(IBM System R -> DB2)*

select v.cru, vt.nom

from vins v, prod p, vitic vt

where v.nv = p.nv and vt.nvt=p.nvt and v.mill = 1983

autre langage : **QUEL** (Ingres)

range of v is vins, range of p is prod, range of vt is vitic

retreive v.cru, vt.nom

where v.nv = p.nv and vt.nvt=p.nvt and v.mill = 1983

Variante : **calcul relationnel de domaine** -> **QBE**

TD

Le langage SQL : Structured Query Language

.Langage de définition des données -LDD- (Création et manipulation des schémas, Gestion d'index, de droits ...)

+

Langage de manipulation de données - LMD- (Interrogations et mises à jour)

+ Embedded SQL

- Langage dérivé de SEQUEL (SYSTEM/R - IBM)
- SQL est une norme depuis 1986
- SQL est proposé dans tous les SGBDs RELATIONNELS (DB2, ORACLE, INGRES, INFORMIX, SYBASE...)

SQL : la recherche d'infos

<u>SYNTAXE GENERALE D'UNE RECHERCHE</u>

|<u>SELECT</u><liste d'attributs projetés>

|<u>FROM</u> <liste des relations touchées par la question>

|<u>WHERE</u><liste de critères de restriction>

|<u>GROUP</u> <u>BY</u><liste des attributs d'agrégation>

|<u>HAVING</u><liste de critères de restriction sur les agrégats>

|<u>ORDER</u> <u>BY</u><liste des attributs de tri du résultat>

Recherche mono-relation

Donner les vins de cru Chablis"

SELECT NV, MIL, DEG

FROM VINS

WHERE CRU = 'Chablis'

Donner la liste de tous les crus, avec élimination des doubles"

SELECT DISTINCT CRU

FROM VINS

Donner tous les vins"

SELECT *

FROM VINS

Donner les numéro et les crus des vins de millésime1976 et de degré 12, triés par ordre roissant sur les crus"

SELECT NV, CRU

FROM VINS

WHERE MIL=1976 AND DEG=12

ORDER BY CRU

Recherche multi-relations

Donner les noms des viticulteurs produisant du Muscadet"

```
SELECT NOM
FROM VITICULTEURS VT, PRODUCTION P, VINS V
WHERE VT.NVT = P.NVT AND P.NV=V.NV
AND V.CRU = 'Muscadet'
```

EXPRESSION DES JOINTURES DE MANIERE PROCEDURALE : UTILISATION DES BLOCS IMBRIQUES

```
SELECT NOM
FROM VITICULTEURS
WHERE NVT  IN  (
        SELECT NVT  FROM PRODUCTION
        WHERE  NV IN (
              SELECT NV FROM VINS
              WHERE  NV IN (SELECT NV FROM VINS)
```

Le cas de l'auto jointure

.Une auto-jointure est une jointure d'une relation avec elle-même.

.L'expression d'une auto-jointure nécessite de déclarer **deux variables sur une même relations**

"Donner les noms des buveurs habitant la même ville"

```
SELECT    B1.NOM, B1.VILLE, B2.NOM
FROM      BUVEURS  B1, BUVEURS  B2
WHERE     B1.VILLE = B2.VILLE
AND       B1.NB <> B2.NB
```

attention : on a ici **deux fois** chaque tuple dans le résultat …

```
AND       B1.NB < B2.NB
```

ne donne qu '**un** tuple

Requêtes complexes en SQL

Select A1, A2…An

From   R1, R2… Rm

where   cond(R1, …, Rm)

Correspond à :

{t : A1…An / ∃ r 1 ∈ R1 … ∃ r m ∈ Rm, (and (t(Ai) = rij(Aj)) and cond (r 1… r m)}

MAIS

L'expression du quantificateur universel ∀ n'est pas triviale en SQL

∀ : division en Alg. Rel.

Exemple

Trouver les buveurs (NB) qui ont commandé des vins millésimé 1992.

$\{ t : NB \ / \quad \exists\, c \in C, \exists\, v \in V, t(NB) = c(NB) \text{ and } c(NV)=v(NV) \text{ and } v(MILL) = 1992 \ ) \}$

```
select  t.NB
from    Buveurs t, Commandes c, Vins v
where  t.NB=c.NB
        and c.NV=v.NV
        and  v.MILL=1992
```

$\forall$ en SQL

Trouver les buveurs (NB) qui ont commandé tous les vins millésimé 1992.

$\{ t : NB \ / \ \forall v \in V,\ v(MILL) = 1992 \Rightarrow \exists c \in C,\ t(NB) = c(NB) \text{ and } c(NV)=v(NV)) \}$

Règles : $\forall x\ f(x)$ équivalent à : not $(\exists x \text{ not } f(x))$

$g(x) \Rightarrow h(x)$ est équivalent à : not $g(x)$ or $h(x)$

not$(g(x) \Rightarrow h(x))$ est équivalent à : $g(x)$ and not $h(x)$

donc

$\{t : NB \ / \ \overline{(}\exists v \in V,\ v(MILL) = 1992 \text{ and } \overline{(}\exists c \in C,\ t(NB) = c(NB) \text{ and } c(NV)=v(NV))) \}$

Conversion en SQL : un niveau d'imbrication pour chaque not ($\exists$ .....)

```
select t.NB from Buveurs t where not exists
   (select * from  Vins v
   where v.MILL = 1992
   and not exists
     (select *  from Commande c where t.NB=c.NB and  c.NV=v.NV))
```

∀ en SQL(suite)

Conversion en SQL : un niveau d'imbrication pour chaque not (∃ .....)

```
select t.NB from Buveurs t where not exists
   (select * from  Vins v
   where v.MILL = 1992
   and not exists
      (select *  from Commande c where t.NB=c.NB and  c.NV=v.NV));
select t.NB from Buveurs t where t.NB not in
   (select  b.NB from  Buveurs b, Vins v
   where v.MILL = 1992
   and b.NB not in
      (select c.NB  from Commande c where  c.NV=v.NV));
select t.NB from Buveurs t where t.NB not in
   (select  b.NB from  Buveurs b, Vins v
   where v.MILL = 1992
   and ( b.NB, v.NV) not in
      (select c.NB, c.NV  from Commande c ));
```

Les prédicats quantifiés
'ANY', 'ALL'

Y a t'il un Chablis plus ancien qu'un Morgon ?"

SELECT * FROM VINS
WHERE CRU = "Chablis"
AND MIL < ANY (SELECT MIL FROM VINS
WHERE CRU = "Morgon")

"Quel est le vin le plus ancien ?"

SELECT NV FROM VINS
WHERE MIL Š ALL (SELECT MIL
FROM VINS)

Recherche sur agrégats

NOTION D'AGREGAT

PARTITIONNEMENT HORIZONTAL D'UNE RELATION SELON DES VALEURS D'ATTRIBUTS, SUIVI D'UN REGROUPEMENT PAR UNE FONCTION DE CALCUL (SOMME, MOYENNE, MIN, MAX, COMPTE, ...)

EXEMPLE:

"Donner, par numéro de vin, la quantité moyenne commandées après le 1/1/88, lorsque la somme des quantités commandées est supérieure à 200"

```
SELECT NV, AVG(QTE), SOM(QTE)
FROM COMMANDE
WHERE DATE    01-01-89
GROUP BY NV
HAVING SOM(QTE) > 200
```

Exemple

COMMANDE	#C	DATE	#V	QTE	#B
	234	12/1/88	30	72	67
	345	2/12/87	43	24	24
	346	16/2/88	30	46	45
	342	23/12/87	30	48	12
	456	3/3/88	43	12	12
	457	10/3/88	30	92	24

WHERE DATE >= 01/01/88

↓

COMMANDE	#C	DATE	#V	QTE	#B
	234	12/1/88	30	72	67
	346	16/2/88	30	46	45
	456	3/3/88	43	12	12
	457	10/3/88	30	92	24

GROUP BY #V

COMMANDE	#C	DATE	#V	QTE	#B
	234	12/1/88	30	72	67
	346	16/2/88	30	46	45
	457	10/3/88	30	92	24
	456	3/3/88	43	12	12

CALCUL DES FONCTIONS

COMMANDE	#V	AVG(QTE)	SOM(QTE)
	30	70	210
	43	12	12

HAVING SOM(QTE) > 200

COMMANDE	#V	AVG(QTE)	SOM(QTE)
	30	70	210

Sémantique de SQL

SELECT <LISTE D'ATTRIBUTS>	PROJECTION
FROM <LISTE DE RELATIONS>	PRODUIT CARTESIEN
WHERE <PREDICAT>	RESTRICTION
GROUP BY <LISTE D'ATTRIBUTS>	REGROUPEMENT
HAVING <PREDICAT>	SELECTION SUR LE REGOUPEMENT
ORDER BY	TRI SUR LE RESULTAT

Les langages relationnels

Les langages théoriques

le calcul relationnel

enjeu = langages assertionnels, puissance d'expression

Les langages internes

l'algèbre relationnelle

enjeu = langages procéduraux, performances

Les langages commerciaux

SQL, mais aussi QBE, L4G…

enjeu = normalisation + interopérabilité, assertionnels + procéduraux

Modèle et langages relationnels

1. Le modèle de données relationnel

Relations

Algèbre relationnelle

Optimisation

2. Le calcul relationnel

Langages de requêtes vs. Langages de programmation

Langages théoriques vs. langages pratiques

Requêtes simples vs. complexes

Consultation vs. Mises à jour

3. Le mécanisme de vues relationnelles

Consultation vs. Mises à jour

Pourquoi des vues externes ?

Un mécanisme de vues répond aux besoins suivants :

- **Adaptation aux applications**
- **Intégration des applications existantes**
- **Dynamique du schéma de la base**
- **Confidentialité et sécurité**
- **Décentralisation de l'administration d'une Base de Données**
- **Réutilisation de requêtes**
- **Hétérogénéité des modèles**

Avec le modèle relationnel et son Langage de Manipulation Assertionnel SQL

UNE VUE = UNE INTERROGATION

Exemples

BASE :

VINS (NV, CRU, ANNEE, DEGRE, NVIT)

VITICULTEURS (NVIT, NOM, PRENOM, VILLE)

BUVEURS (NB, NOM, PRENOM, VILLE)

COMMANDES (DATE, NB, NV, QTE)

On peut définir de nombreuses applications particulières qui n'utilisent qu'une partie des données

Par exemple, le rapport entre les consommations et le lieu d'habitation des buveurs en 1989

VUE = Influence géographique sur la consommation de l'année 1989

Par simple restriction

BUVEURS-1 (NB, VILLE)

ACHAT-1(NB, NV, QTE, DATE)

VINS-1 (NV, CRU)

Par restructuration

ACHAT-2(NB, VILLE, CRU, QTE, DATE)

Par aggrégation

BUVEURS-3 (NB, VILLE)

ACHAT-3 (NB, NV, QTE)

VINS-3 (NV, CRU)

Par combinaison

CPV(VILLE, CRU, QTE)

VUE = Influence géographique sur la consommation de l'année 1989

Par simple restriction

BUVEURS-1 (NB, VILLE)
ACHAT-1(NB, NV, QTE, DATE)
VINS-1 (NV, CRU)

```
CREATE VIEW BUVEURS-1 AS
    SELECT NB, VILLE
    FROM BUVEURS
```

```
CREATE VIEW ACHAT-1 AS
    SELECT NB, NV, QTE, DATE
    FROM COMMANDES
    WHERE DATE    "890101"
      AND DATE Š "891231"
```

```
CREATE VIEW VINS-1 AS
    SELECT NV, CRU
    FROM VINS
```

VUE = Influence géographique sur la consommation de l'année 1989

```
CREATE VIEW ACHAT-2 AS
        SELECT NB, VILLE, CRU; QTE, DATE
        FROM BUVEURS, COMMANDES, VINS
        WHERE BUVEURS.NB = COMMANDES.NB
        AND COMMANDES.NV = VINS.NV
        AND DAT E  "890101"
        AND DATE Š "891231"
```

Par restructuration

ACHAT-2(NB, VILLE, CRU, QTE, DATE)

VUE = Influence géographique sur la consommation de l'année 1989

```
CREATE VIEW ACHAT-3 (NB, NV, QTE) AS
    SELECT NB, NV, SUM(QTE)
    FROM COMMANDES
    WHERE DATE     "890101"
    AND DATE Š "891231"
    GROUP BY NB, NV
```

Par aggrégation

BUVEURS-3 (NB, VILLE)
ACHAT-3 (NB, NV, QTE)
VINS-3 (NV, CRU)

VUE = Influence géographique sur la consommation de l'année 1989

```
CREATE VIEW CPV (VILLE, CRU, QTE) AS
        SELECT VILLE, CRU, SUM(QTE)
        FROM BUVEURS, COMMANDES, VINS
        WHERE BUVEURS.NB = COMMANDES.NB
        AND COMMANDES.NV = VINS.NV
        AND DAT E "890101"
        AND DATE Š "891231"
        GROUP BY VILLE, CRU
```

Par combinaison

CPV (VILLE, CRU, QTE)

Qu'est ce qu'une vue externe ?

- Une relation correspondant à la **vision d'un groupe d'utilisateurs**

- Une **relation virtuelle** ne contenant pas de tuples

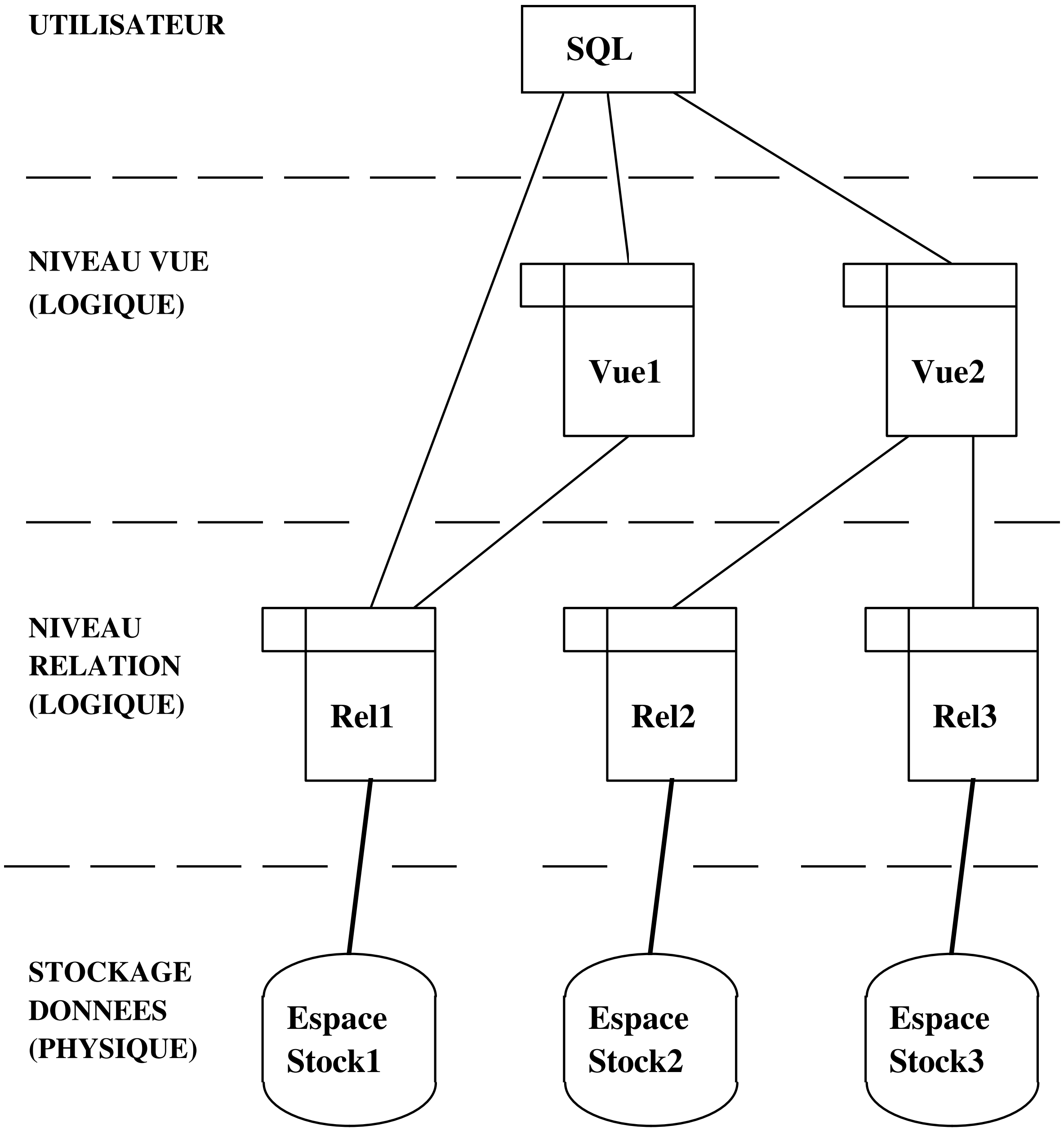
- Une relation définie à partir des relations de base (contenant des tuples) et/ou d'autres vues

- Une **relation manipulable**

  comme les autres relations pour l'interrogation

  avec certaines **restrictions** pour les mises à jour

Vues externes



Simplicité des méthodes avec le modèle relationnel

Modification de question

Implantée dans le système INGRES

Interprétation à l'exécution

Implantée dans SYSTEM/R

Concrétisation de vues

Implantée dans SABRINA

Conclusion : SQL