Systèmes de Gestion de Bases de données (SGBD)
http://www.bd.enst.fr/

Structured Query Language

- LMD
- LDD
- Vues

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)

- 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

SYNTAXE GENERALE D'UNE RECHERCHE

|SELECT<liste d'attributs projetés>

|FROM <liste des relations touchées par la question>

|WHERE<liste de critères de restriction>

|GROUP BY<liste des attributs d'agrégation>

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

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

Recherche mono-relation

Donner les vins de cru Chablis"

<u>SELECT</u> NV, MIL, DEG

<u>FROM</u> VINS

<u>WHERE</u> CRU = 'Chablis'

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

<u>SELECT</u> <u>DISTINCT</u> CRU

<u>FROM</u> VINS

Donner tous les vins"

<u>SELECT</u> *

<u>FROM</u> VINS

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

<u>SELECT</u> NV, CRU

<u>FROM</u> VINS

<u>WHERE</u> MIL=1976 <u>AND</u> DEG=12

<u>ORDER</u> <u>BY</u> 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 VT
WHERE EXISTS (
        SELECT * FROM PRODUCTION P
        WHERE P.NVT=VT.NVT AND EXISTS (
            SELECT * FROM VINS V
            WHERE V.NV = P.NV AND CRU = 'Muscadet')
        )
```

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  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
```

/* **mieux** : on a ici **une seule fois** chaque tuple dans le résultat */

Requêtes complexes en SQL

Select A1, A2…An

From R1, R2… Rm

where cond(R1, …, Rm)

Correspond à :

$\{t : A1…An \; / \; \exists\, r\,1 \in R1 \,…\, \exists\, r\,m \in Rm, (\text{and } (t(Ai) = rij(Aj))\text{ and cond }(r\,1…\,r\,m)\}$

MAIS

L'expression du quantificateur universel $\forall$ n'est pas triviale en SQL

$\forall$: division en Alg. Rel.

Exemple

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

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

```
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 • x(not• f (x))

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

donc 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 $\overline{\exists}$ ..., (...)

```
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 : not exists / not in

```
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")
```

ou bien

```
SELECT * FROM VINS V1
WHERE V1.CRU = "Chablis"
AND  EXISTS  (SELECT * FROM VINS V2
     WHERE V2.MIL > V1.MIL AND V2.CRU =
     "Morgon")
```

Y a t'il un Chablis plus ancien que tous les Morgon ?

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

ou bien

```
SELECT * FROM VINS V1
WHERE V1.CRU = "Chablis"
AND  NOT EXISTS  (SELECT * FROM VINS V2
     WHERE V2.MIL < V1.MIL AND V2.CRU =
     "Morgon")
```

Quel est le vin le plus ancien ?

```
SELECT NV FROM VINS
WHERE MIL >= ALL (SELECT MIL
                  FROM VINS)
```

ou bien

```
SELECT NV FROM VINS
WHERE MIL = (SELECT MAX(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 - SUM, MOYENNE - AVG, MIN, MAX, COMPTE - COUNT, ...)

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), SUM(QTE)
FROM COMMANDE
WHERE DATE > '01-01-88'
GROUP BY NV
HAVING SUM(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)	SUM(QTE)
	30	70	210
	43	12	12

HAVING SUM(QTE) > 200

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

Fonctions sur tuples v. sur groupes de tuples

Ne pas confondre

- attributs calculés par tuple:

  fonctions compatibles avec les types/domaines des attributs, ex:

```
SELECT 0.9*DEG                                  /* multiplication */
   FROM CRU WHERE NV = 33;
SELECT NB,  PRENOM|| ' ' ||NOM                  /* concaténation */
   FROM BUVEURS;
SELECT NV, TO_CHAR(DAT, ' YYYY-MM-DD ') , QTE, NB /* conversion */
   FROM COMMANDE;
```

- attributs calculés par groupes de tuples:

```
SELECT NV, MIN(DAT), MAX(DAT), SUM(QTE), COUNT(NB)
      FROM COMMANDE
      GROUP BY NV
```

Sémantique du SELECT

5- SELECT <LISTE D'ATTRIBUTS>	PROJECTION
1- FROM <LISTE DE RELATIONS>	PRODUIT CARTESIEN
2- WHERE <PREDICAT>	RESTRICTION
3- GROUP BY <LISTE D'ATTRIBUTS>	REGROUPEMENT
4- HAVING <PREDICAT>	SELECTION SUR LE REGOUPEMENT
6- ORDER BY	TRI SUR LE RESULTAT

Mises à jour en SQL

• Ajout de tuples:

INSERT INTO **<RELATION>** VALUES (**<LISTE DE VALEURS>**)

ou

INSERT INTO **<RELATION>** SELECT **<LISTE D'ATTRIBUTS>**
FROM **<LISTE DE RELATIONS>** WHERE **<PREDICAT>**

• Suppression de tuples:

DELETE FROM **<RELATION>** WHERE **<PREDICAT>**

si pas de clause where alors suppression de tous les tuples!

• Modification de tuples:

UPDATE **<RELATION>**

SET **<LISTE DE PAIRES (ATTRIBUT=VALEUR)>**

WHERE **<PREDICAT>**

Schéma d'une relation

1 . CREATION D'UN SCHEMA DE RELATION

CREATE **TABLE** VIN (NV Integer, CRU Char(20), MIL Integer);

2 . AJOUT D'UN ATTRIBUT "A DROITE"

ALTER TABLE VIN
    ADD DEG Integer;

3 . SUPPRESSION D'UN SCHEMA DE RELATION

DROP **TABLE** VIN;

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 s 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 s 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 >= ‘01-01-89’
          AND DATE =< ‘31-12-89’
```

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

Implantation des vues avec le modèle relationnel

Modification de question

Implantée dans le système INGRES

Interprétation à l'exécution

Implantée dans SYSTEM/R

Matérialisation de vues

Implantée dans SABRINA

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

Par restructuration

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

```
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 DATE >= "890101"
    AND DATE =< "891231"
```

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

Par aggrégation

BUVEURS-3 (NB, VILLE)

ACHAT-3 (NB, NV, QTE)

VINS-3 (NV, CRU)

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

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

Par combinaison

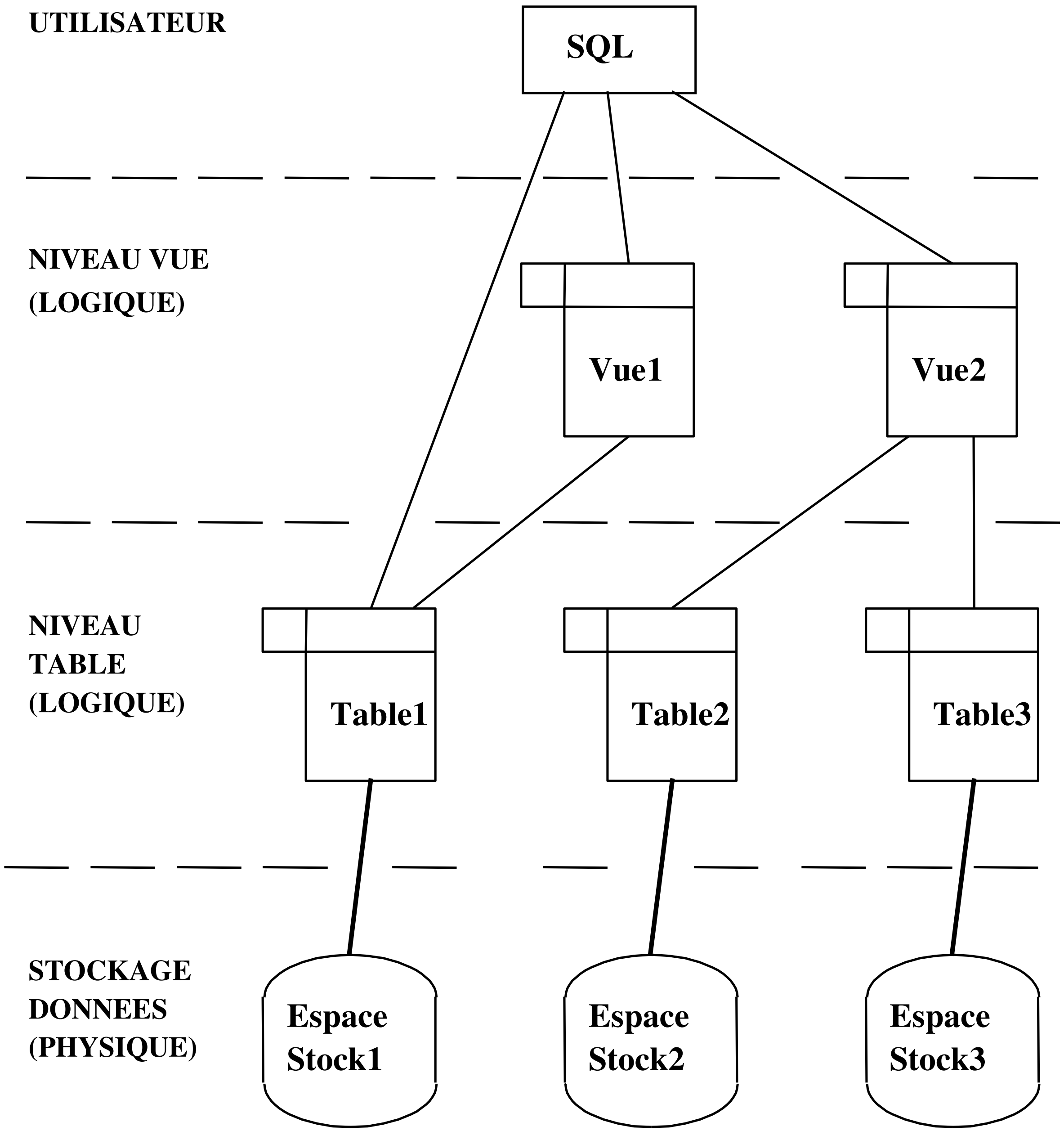
CPV (VILLE, CRU, QTE)

```
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 DATE BETWEEN ‘01-01-89’ AND ‘31-12-89’
    GROUP BY VILLE, CRU
```

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



Conclusion : SQL