

# Base de données

---

Séance 5

Langage de Manipulation de Données :

INSERT, UPDATE, DELETE

Introduction à l'Algèbre relationnelle

# SQL : Langage de Manipulation des Données (LMD)

---

- Le LDD :
  - permet d'effectuer des opérations sur les structures
- Le LMD
  - permet d'effectuer des opérations sur les données
  - concerne le cycle de vie des données dans une application
  - implémente le modèle «CRUDE» (Create, Read, Update, Delete).

# L'implémentation du CRUDE en SQL

---

- Le CRUDE en SQL est constitué des quatre ordres :
  - INSERT : insertion d'enregistrement(s)
  - UPDATE : modification d'enregistrement(s)
  - DELETE : suppression d'enregistrement(s)
  - SELECT : accès aux enregistrement(s)

# LMD : tables pour les exemples

---

```
CREATE TABLE employe (  
    nom VARCHAR2(30),  
    prenom VARCHAR2(30),  
    date_embauche DATE);
```

```
CREATE TABLE etudiant (  
    nom VARCHAR2(30),  
    prenom VARCHAR2(30),  
    classe CHAR(4) DEFAULT 'ING1');
```

# INSERT : ajout d'une ligne

---

- Syntaxe simplifiée :

```
INSERT INTO <nom_table>  
VALUES (<expr>[,<expr>...]);
```

- Exemple :

```
INSERT INTO employe  
VALUES ('DERAY', 'ODILE',  
to_date('01/09/1972', 'DD/MM/YYYY')) ;
```

- Attention à l'ordre des colonnes !

# INSERT : ajout d'une ligne

---

□ Syntaxe INSERT explicite (plus sûr) :

```
INSERT INTO <nom_table> (<col1> [, ... ])  
VALUES (<expr1> [, ... ]);
```

□ Exemple :

```
INSERT INTO employe  
    (nom, prenom, date_embauche)  
VALUES ('DERAY', 'ODILE',  
    to_date('01/09/1972', 'DD/MM/YYYY'));
```

# INSERT : insertion partielle

---

- ❑ N-uplet partiel complété par des valeurs NULL ou par défaut.
- ❑ Complétion explicite :

```
INSERT INTO employe  
VALUES ('DERAY', 'ODILE', NULL);  
  
CREATE TABLE etudiant (...,  
classe CHAR(4) DEFAULT 'ING1');  
INSERT INTO etudiant  
VALUES ('DUPONT', 'EMILE', DEFAULT);
```



# INSERT : insertion partielle

---

## □ Complétion implicite :

- avec la valeur NULL

```
INSERT INTO employe (nom, prenom)
VALUES ('DERAY', 'ODILE');
```

- avec la valeur par défaut

```
INSERT INTO etudiant (nom, prenom)
VALUES ('DUPONT', 'EMILE');
```

- toutes les valeurs par défaut

```
INSERT INTO ma_table DEFAULT VALUES;
```

# INSERT : insertion multiple

---

- Insérer le résultat d'un SELECT

```
INSERT INTO personne (nom, prenom)  
SELECT nom, prenom FROM etudiant;
```

- Ajustements possibles
  - Insertion conditionnelle
  - Répartition sur plusieurs tables

# INSERT : contraintes

---

- Attention aux respects des contraintes !
  - Clés primaires et UNIQUE (pas de doublons)
  - Clés étrangères (existence)
  - CHECK, NOT NULL

# UPDATE : modification des données

---

- Mettre à jour les n-uplets d'une table vérifiant un certain prédicat :

```
UPDATE <nom_table>  
SET <col>=<expr>[,<col>=<expr>...]  
[WHERE <predicat>];
```

- Les expressions sont évaluées pour chaque n-uplets vérifiant le prédicat.
- Si aucune clause WHERE n'est spécifiée, **tous** les enregistrements de la table sont modifiés !

# UPDATE : exemple

---

**UPDATE**

**personne**

**SET**

**date\_naissance = DATE '1900-01-01'**

**WHERE**

**date\_naissance IS NULL;**

# DELETE : suppression des données

---

- ❑ Supprime d'une table les n-uplets qui vérifient un certain prédicat.
- ❑ Syntaxe générale :

```
DELETE FROM <tab> [WHERE <predicat>];
```

- ❑ Si aucune clause WHERE n'est spécifiée, **tous** les tuples sont supprimés (cf. update) !

# DELETE : exemples

---

```
DELETE FROM personne;
```

```
DELETE FROM personne  
WHERE date_naissance IS NULL;
```

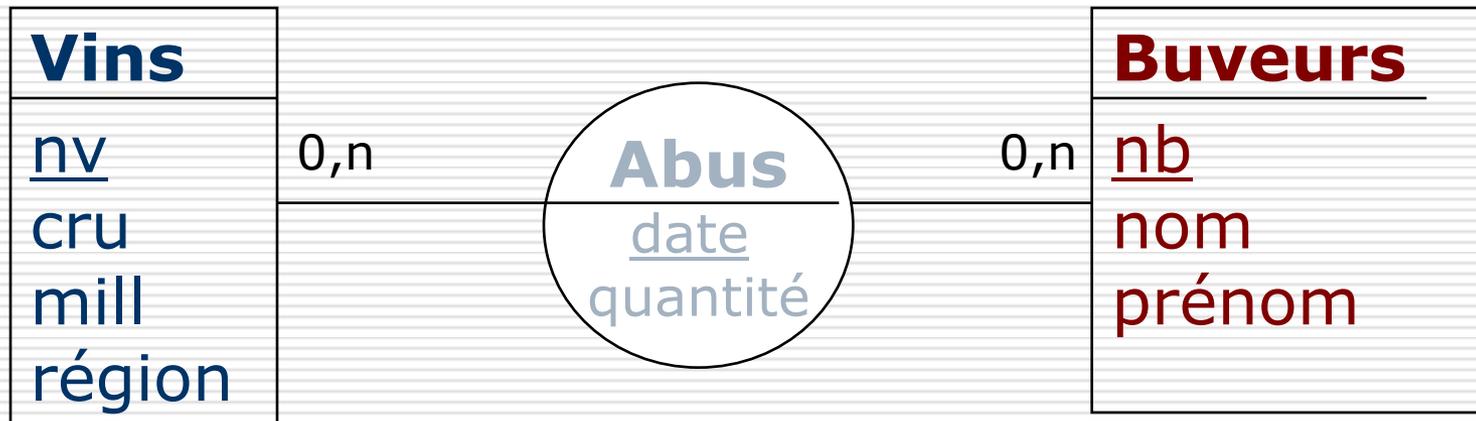
# Algèbre relationnelle

---

- Introduction
  - Opérateurs unaires
    1. Projection
    2. Restriction
    3. Tri
  - Requête et arbre algébrique
-

# Exemple MCD

---



# Exemple : schéma relationnel

---

- Vins(nv : INT, cru : TEXT, mill : INT, region : TEXT)  
Buveurs(nb : INT, nom : TEXT, prenom : TEXT)  
Abus(nb : INT, nv : INT, dateC : DATE, quantite : INT)
  - Clés étrangères :  
Abus.nv référence Vins.nv  
Abus.nb référence Buveurs.nb
-

# Exemple de relation

---

VINS	Cru	Mill	Région
	CHENAS	1983	BEAUJOLAIS
	TOKAY	1980	ALSACE
	TAVEL	1986	RHONE
	CHABLIS	1986	BOURGOGNE
	ST-EMILION	1987	BORDELAIS



*L'abus d'alcool est dangereux pour la santé.  
A consommer avec modération ☺*

---

# Principe des langages algébriques

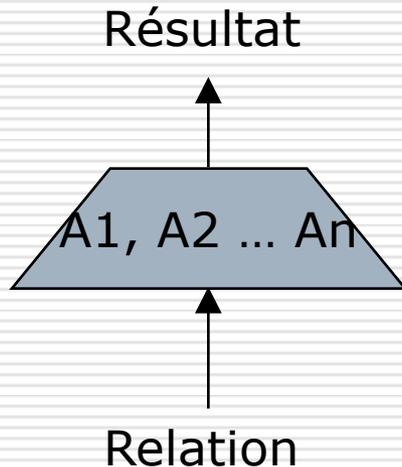
---

- L'information résultat peut être mise sous forme de relation.
  - La relation résultat est obtenue en appliquant successivement des opérateurs binaires ou unaires sur les relations de la base ou sur des relations de travail intermédiaires.
-

# Projection

---

- Opération sur une relation R consistant à composer une relation résultat R1
  - en enlevant à la relation initiale R tous les attributs non mentionnés en opérandes et
  - en éliminant les enregistrements en double qui sont conservés une seule fois.



$$\text{Résultat} = \Pi_{A_1, A_2 \dots A_n}(\text{Relation})$$

---

# Exemple

---

VINS	Cru	Mill	Région
	VOLNAY	1983	BOURGOGNE
	VOLNAY	1979	BOURGOGNE
	CHENAS	1983	BEAUJOLAIS
	JULIENAS	1986	BEAUJOLAIS

$\Pi_{\text{cru,région}}$

$\Pi(\text{VINS})$	Cru	Région
	VOLNAY	BOURGOGNE
	VOLNAY	BOURGOGNE
	CHENAS	BEAUJOLAIS
	JULIENAS	BEAUJOLAIS

---

# Exemple

---

VINS	Cru	Mill	Région
	VOLNAY	1983	BOURGOGNE
	VOLNAY	1979	BOURGOGNE
	CHENAS	1983	BEAUJOLAIS
	JULIENAS	1986	BEAUJOLAIS

$\Pi_{\text{cru,région}}$

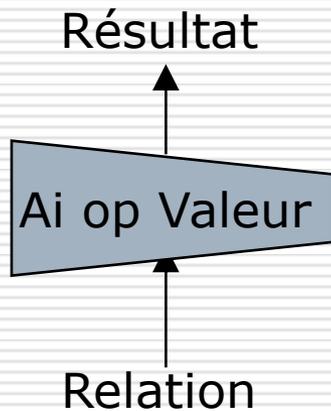
$\Pi(\text{VINS})$	Cru	Région
	VOLNAY	BOURGOGNE
	<del>VOLNAY</del>	<del>BOURGOGNE</del>
	CHENAS	BEAUJOLAIS
	JULIENAS	BEAUJOLAIS

---

# Restriction

---

- Opération sur une relation R produisant une relation résultat R1
  - de même schéma
  - mais comportant les seuls enregistrements qui vérifient la condition précisée en argument.



$$\text{Résultat} = \sigma_{\text{Ai op Valeur}}(\text{Relation})$$

- Critère de qualification avec **op** dans  $\{=, <, \leq, >, \geq, \neq\}$
  - Combinaison de critères simples : "and", "or" et "not"
-

# Exemple

---

VINS	Cru	Mill	Région
	VOLNAY	1983	BOURGOGNE
	VOLNAY	1979	BOURGOGNE
	CHENAS	1983	BEAUJOLAIS
	JULIENAS	1986	BEAUJOLAIS

$\sigma_{\text{MILL} > 1983}$



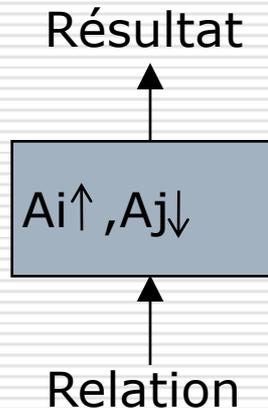
$\sigma$ (VINS)	Cru	Mill	Région
	JULIENAS	1986	BEAUJOLAIS

---

# Tri

---

- Opération sur une relation R produisant une relation R1
  - de même schéma
  - contenant tous les enregistrements de R réordonnés suivant les critères en argument.



$$\text{Résultat} = \text{Tri}(\text{Relation}, A_i \uparrow, A_j \downarrow)$$

- Par défaut : ordre croissant
-

# Exemple

---

VINS	Cru	Mill	Région
	VOLNAY	1983	BOURGOGNE
	VOLNAY	1979	BOURGOGNE
	CHENAS	1983	BEAUJOLAIS
	JULIENAS	1986	BEAUJOLAIS

**Tri(VINS, année↓, cru↑)**

Tri(VINS)	Cru	Mill	Région
	JULIENAS	1986	BEAUJOLAIS
	CHENAS	1983	BEAUJOLAIS
	VOLNAY	1983	BOURGOGNE
	VOLNAY	1979	BOURGOGNE

---

# Arbre algébrique

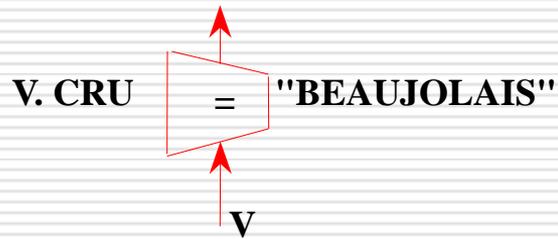
---

- Arbre représentant une requête dont
    - les nœuds terminaux représentent les relations,
    - les nœuds intermédiaires des opérations de l'algèbre relationnelle,
    - le nœud racine le résultat de la requête,
    - et les arcs les flux de données entre les opérations.
-

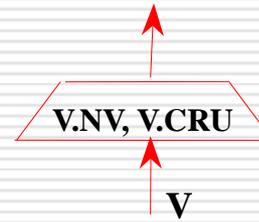
# Opérateurs algébriques

---

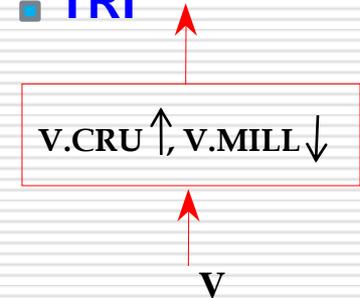
## ■ RESTRICTION



## ■ PROJECTION



## ■ TRI



# Arbre algébrique : exemple 1

---

□ Donner les **degrés** des **vins** de **crus** Morgon et de **millésime 1978**.

■  $R1 = \sigma_{\text{cru} = \text{'Morgon'}} (\text{VINS})$

$R2 = \sigma_{\text{mill} = 1978} (R1)$

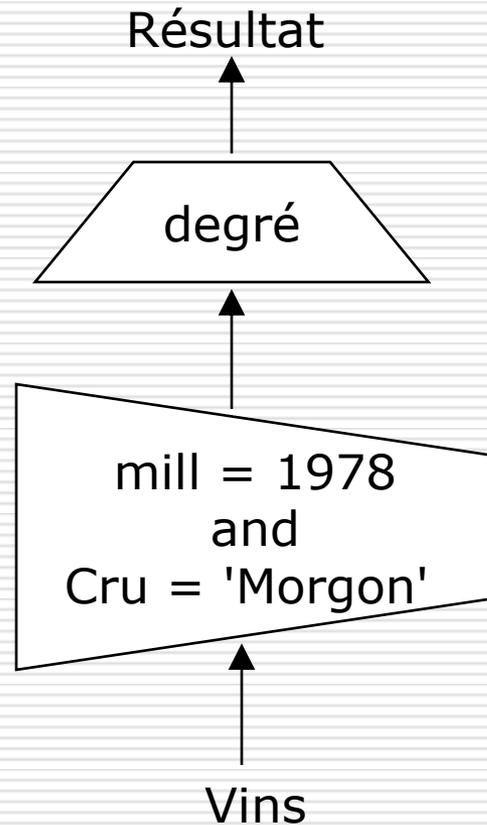
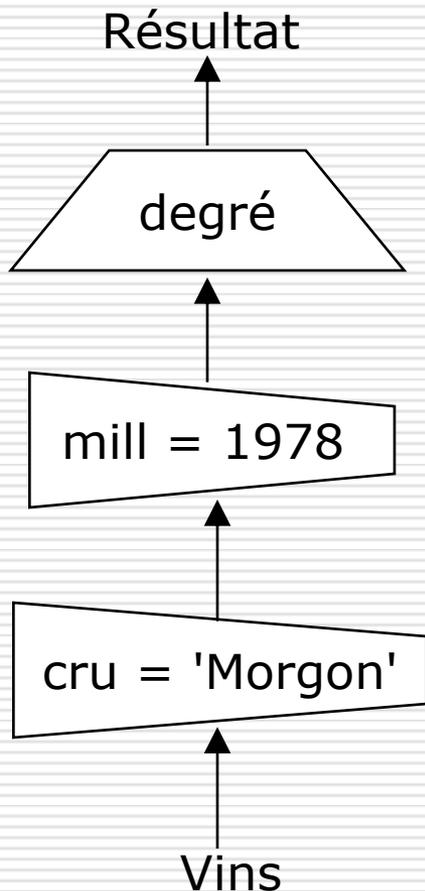
Résultat =  $\Pi_{\text{degré}} (R2)$

■ Résultat =  $\Pi_{\text{degré}} (\sigma_{\text{mill} = 1978 \text{ and cru} = \text{'Morgon'}} (\text{VINS}))$

---

# Arbre algébrique : exemple 1

---



# Arbre algébrique : exemple 2

---

- Donner les crus et degrés des vins Beaujolais, triés par degré décroissant.
  
  - $R1 = \sigma_{\text{region} = \text{'Beaujolais'}} (\text{VINS})$   
 $R2 = \Pi_{\text{cru,degré}}(R1)$   
Résultat = Tri( $R2, \text{degré} \downarrow$ )
  
  - Résultat = Tri( $\Pi_{\text{cru,degré}} (\sigma_{\text{region} = \text{'Beaujolais'}} (\text{VINS}))$ ,  
 $\text{degré} \downarrow$ )
-

# Arbre algébrique : exemple 2

---

