IFT 6243

Concepts
de
Bases de données avancées

Claude Frasson

A 2014

Université de Montréal
Département IRO

IFT 6243 Concepts de Bases de données avancées

Claude Frasson

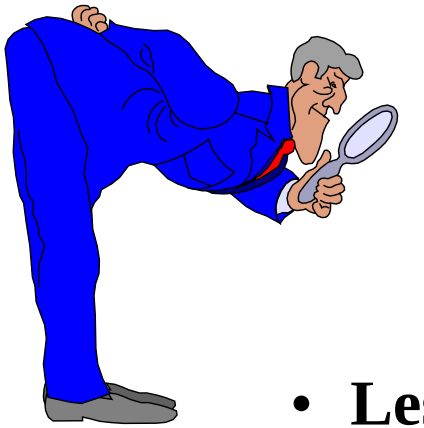
Objectifs : Le cours met l'accent sur les approches intelligentes que l 'on retrouve dans de nombreux systèmes comme les bases de données, les agents intelligents ou les systèmes tutoriels intelligents. Le cours est appuyé par des projets qui feront l 'objet d 'exposés et de rapports sous forme d 'articles. On examinera successivement :

	nombre approximatif de semaines
• L'évolution des systèmes de bases de données	1
• Les langages SQL, QBE, QPE	2
• Les agents intelligents	4
• Les systèmes tutoriels intelligents	4
• Les bases d'images, géographiques et multimédia	2
• Les bases de données intelligentes	1

- **Bibliographie**

  - K. Parsaye, M. Chignell : Intelligent Database Tools and Applications, Wiley.
  - C. Frasson, G. Gauthier, A. Lesgold : Intelligent Tutoring Systems, LNCS vol 1086, Springer Verlag, 1996.
  - C. Frasson, G. Gauthier, K. Van Lehn : Intelligent Tutoring Systems, LNCS vol 1839, Springer Verlag, 2000.
  - Korth, Silberschatz : Database systems concepts, Mc GrawHill
  - J. Muller, M. Wooldridge, N. Jennings : Intelligent Agents 3, Agent Theories, Architectures and Languages, Springer, 1996.
  - N. Jennings, M. Wooldridge : Agent Technology, Foundations, Applications, and Markets, Springer, 1997.

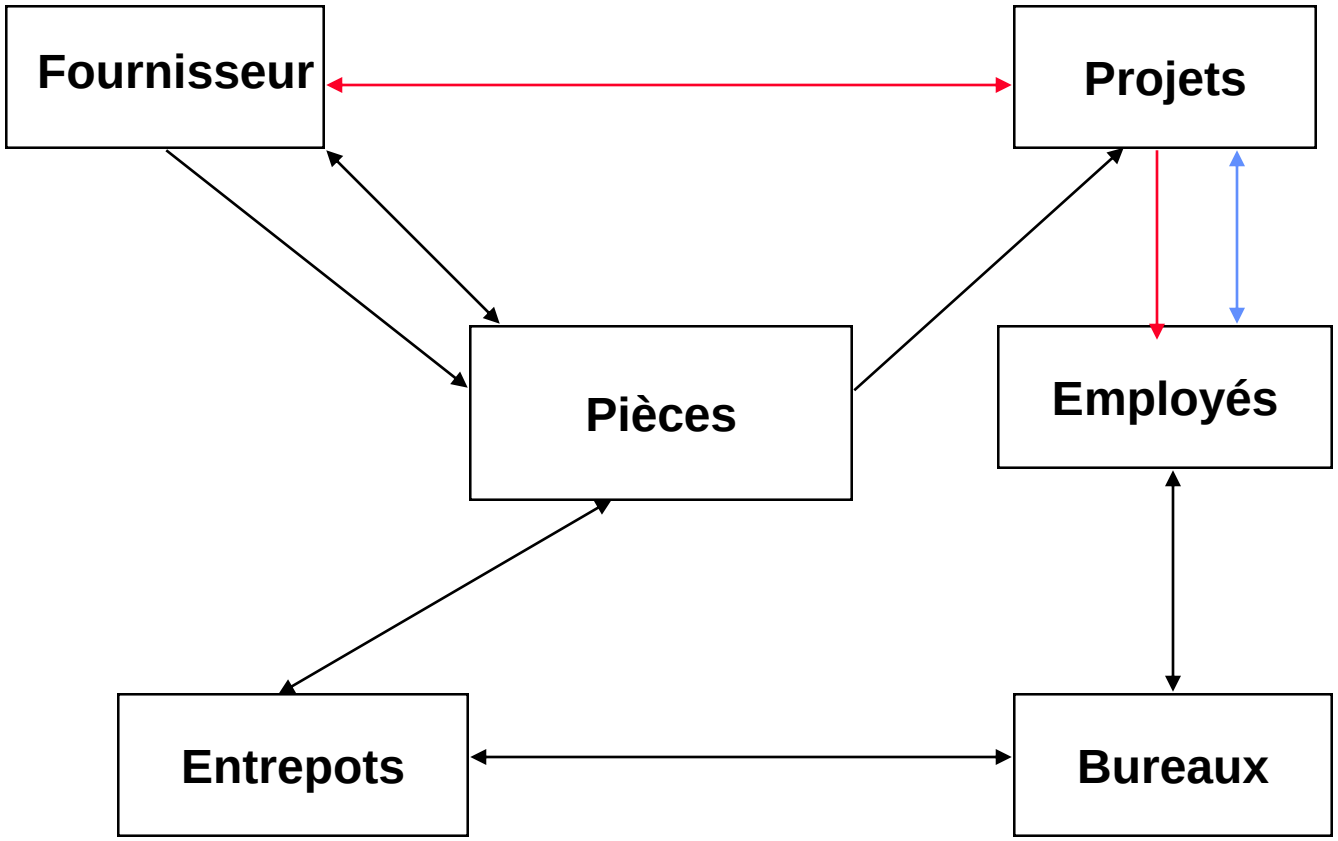
L ’évolution des SGBD



Objectifs d'apprentissage

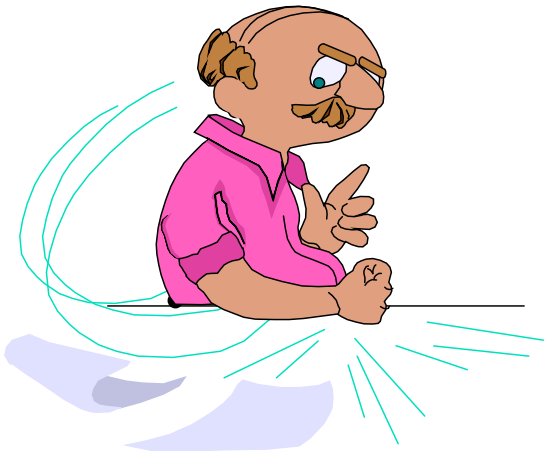
- **Les notes suivantes sont supposées connues et serviront de référence pour ceux qui n ont pas suivi ce cours.**
- **Il s’agit, en parcourant rapidement ces notes,**
- **De comprendre l’évolution des SGBD**
- **De connaître les objectifs d'un SGBD**
- **De connaître le fonctionnement d'un SGBD**
- **De distinguer les évolutions des SGBD**

Le concept de bases de données



Base de données : ensemble de données et de liens qui les unissent

Indépendance des données

- **notion de dépendance ---> la facon dont les données sont organisées et la manière dont on les obtient sont imposées par l'application et la connaissance de cette organisation se répercute sur la logique même de l'application**
- 

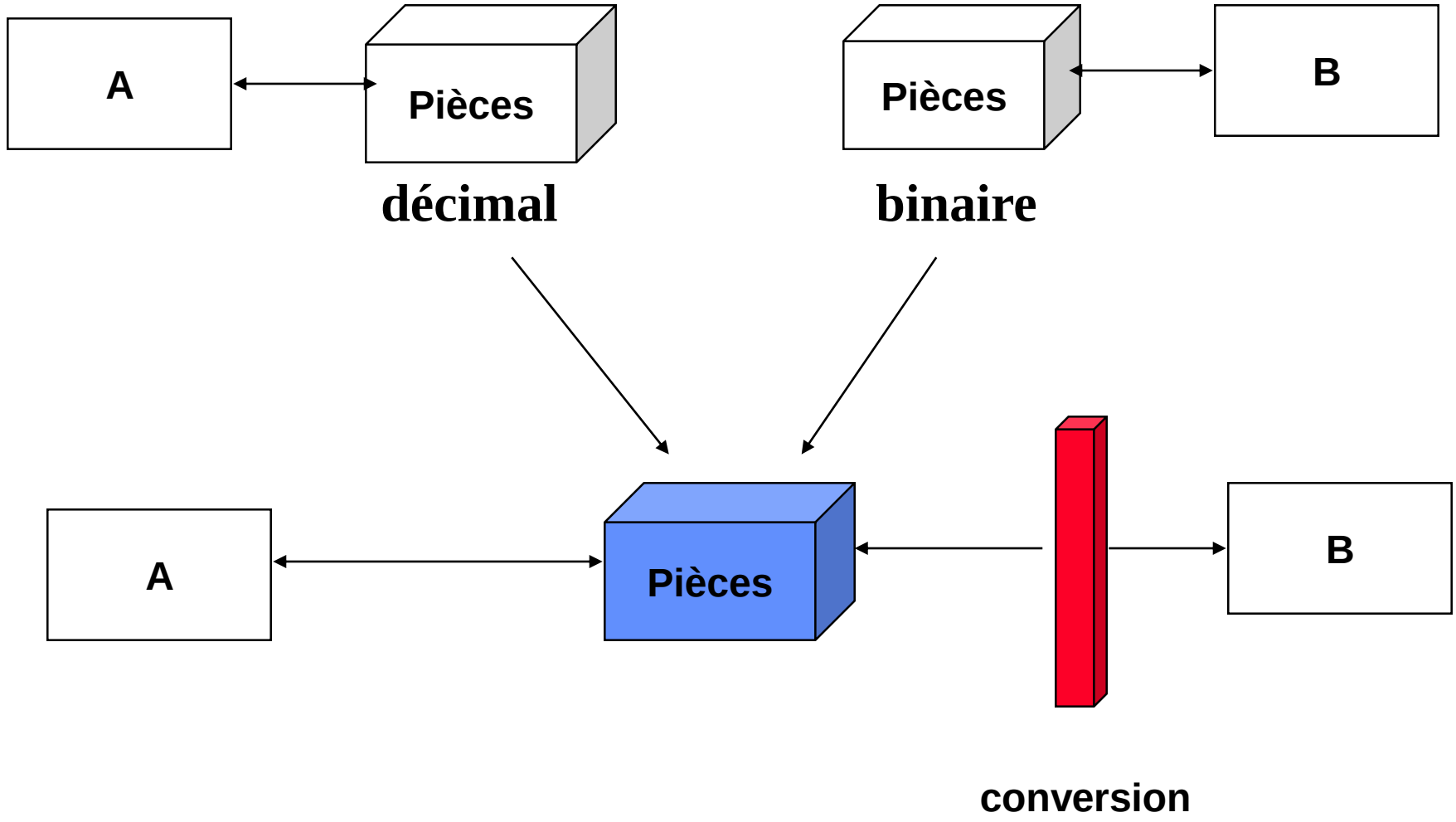
  Impossible de changer

  la structure de stockage

  la méthode d'accés

  sans entrainer de grandes modifications
- **Un système de bases de données doit éviter cette dépendance et ceci pour deux raisons**

⇒**Différentes applications nécessitent parfois des vues différentes des mêmes données**



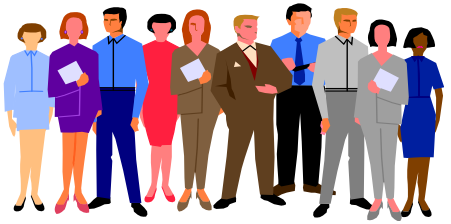
⇒ **L'administrateur doit avoir la liberté de changer la structure de stockage ou la stratégie d'accés selon la nécessité**
Raisons :

- **nouvelles normes**
- **nouvelles priorités des applications**
- **nouvelles unités de stockage**
- **...**

- **les approches de programmation classique entraînent une perte de temps de l'activité des programmeurs (dévolué à la production) en maintenance**

- **L'indépendance constitue un des objectifs principaux des systèmes de bases de données. Elle est définie par l'immunité des applications vis-a vis des changements dans**
  - **la structure de stockage**
  - **la stratégie d'accés**

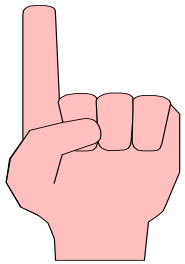
Objectifs d'un SGBD



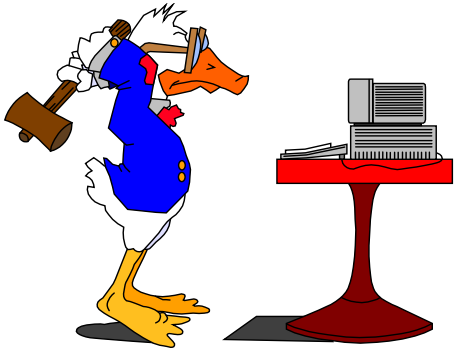
- possibilité d'avoir différentes vues

- investissement intellectuel protégé

- Clarté

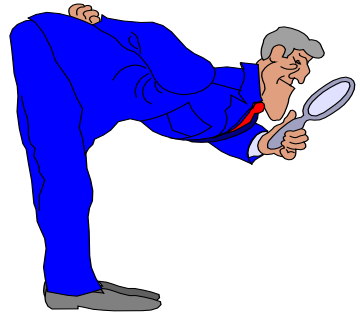


- facilité d'utilisation
- résolution rapide de requêtes aléatoires

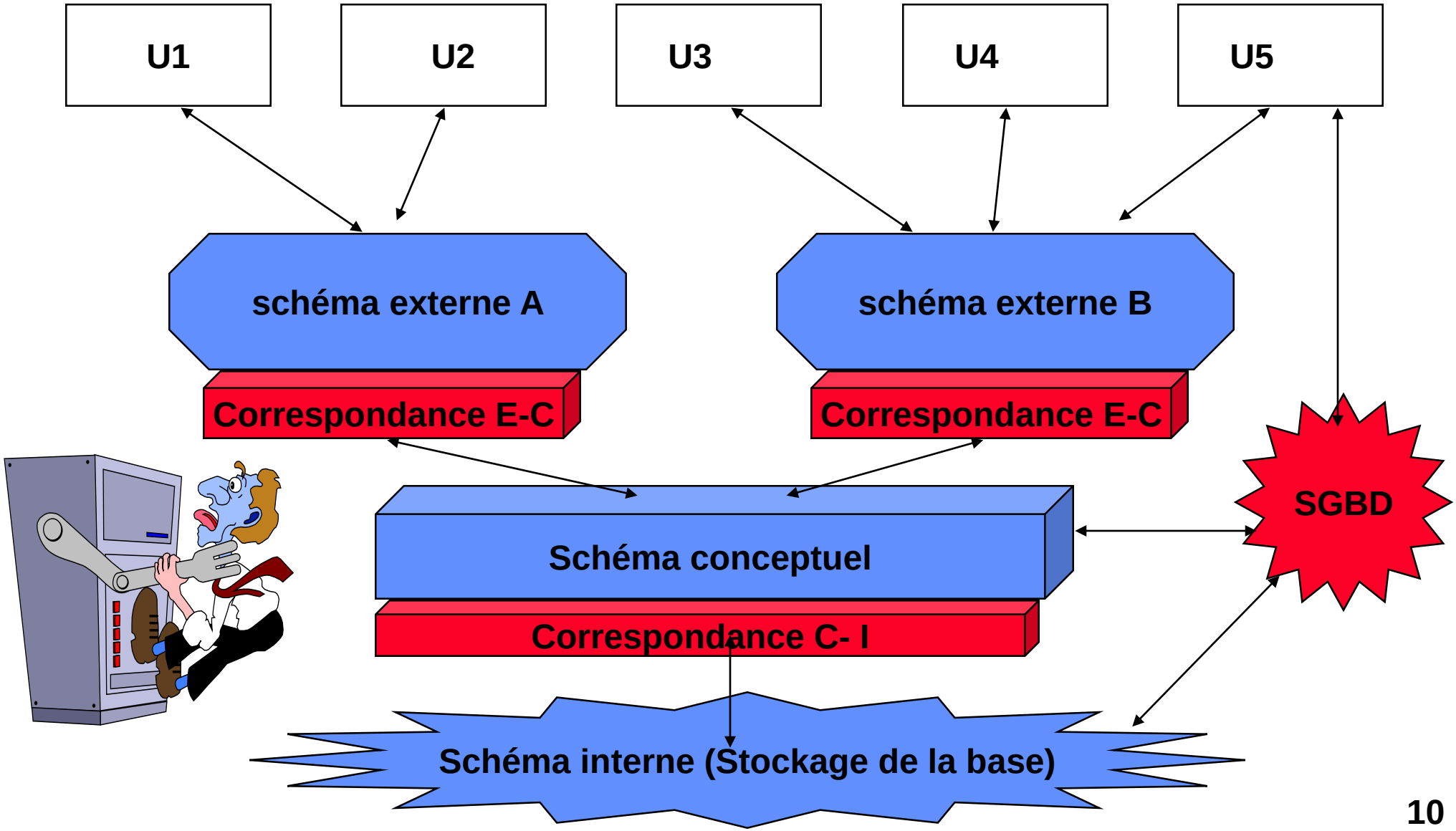


- performances acceptables

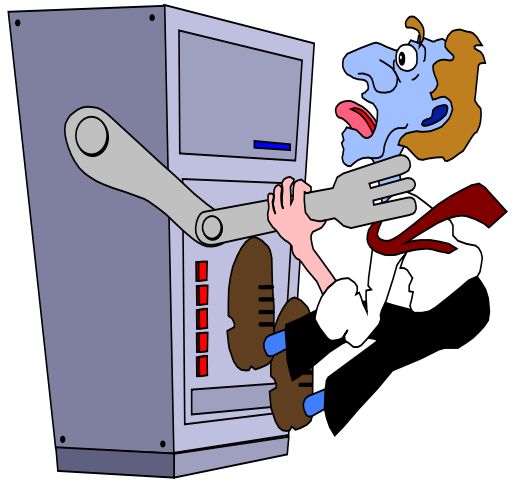
- **confidentialité**
- **sécurité**
- **indépendance physique**
- **indépendance logique**
- **redondance contrôlée**
- **dictionnaire de données**
- **interface de programmation de haut niveau**
- **langage orienté vers l'utilisateur final**
- **controle d'intégrité**
- **reprise aisée en cas de pannes**
- **outils d'aide à la conception**
  à l'optimisation



Architecture d'un système de bases de données

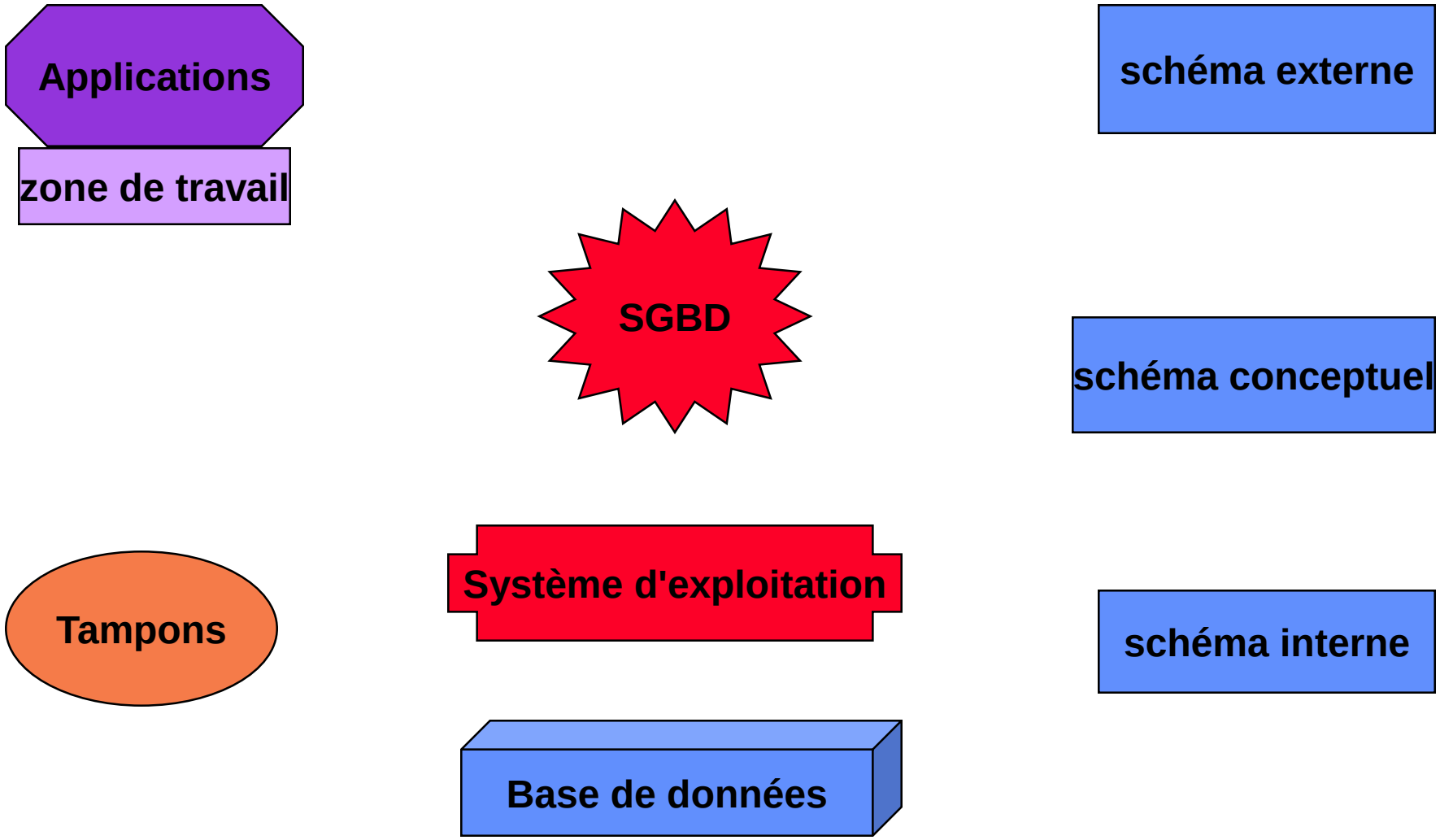


Administrateur de la base



- **Responsable de l'organisation du système et de la base de facon à obtenir les meilleures performances pour l'entreprise**
- **personne rare et chère**
- **décide du contenu de la base**
- **décrit le modèle de données dans le LDD**
- **décide de la représentation des données dans la base**
- **écrit la correspondance Modèle-Stockage**
- **structure de stockage, stratégie d'accès**
- **définit les autorisations et les procédures de validation**
- **définit une procédure de sauvegarde**

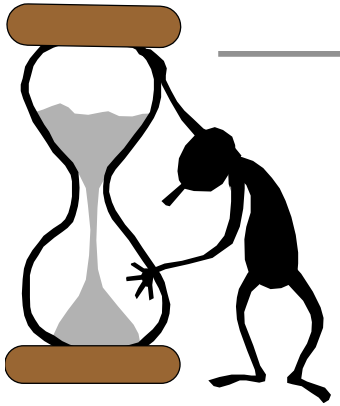
Fonctionnement d'un SGBD



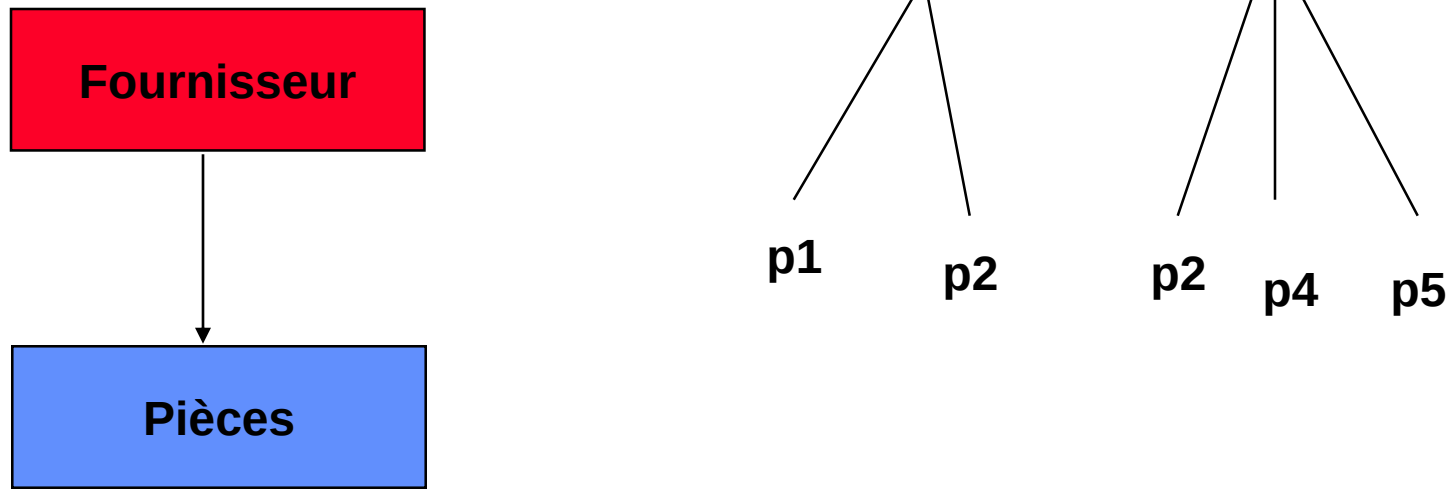
Historique des bases de données

- **Modèle hiérarchique**
- **Modèle réseau**
- **Modèle relationnel**
- **Modèle entité-relation**
- **Modèle objet**

Les différents modèles de SGBD



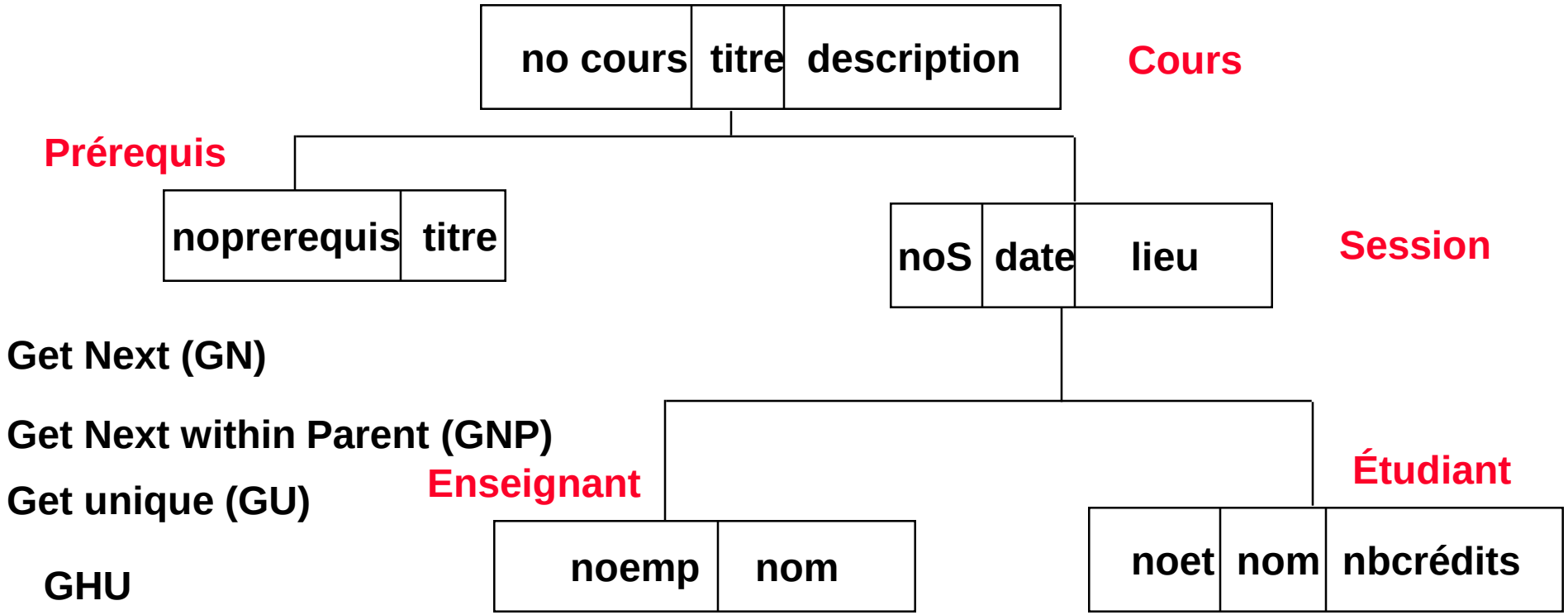
Le modèle hiérarchique



- **Inconvénients**
  - **modèle à bien choisir (performances)**
  - **Quels sont les fournisseurs qui fournissent la pièce P2 ?**
  - **Quelles sont les pièces fournies par le fournisseur F5 ?**
  - **Programmes compliqués (avec les niveaux)**
- **Anomalies**
  - **relation 1: plusieurs**
  - **ajout : il n'est pas possible d'ajouter une pièce tant que son fournisseur n'existe pas**
  - **suppression : si on supprime un fournisseur on supprime toutes ses pièces (inconvénient si unique)**
  - **-mise à jour : mise à jour de la couleur de la pièce P2**
    recherche partout dans toute la base
- **Système IMS**

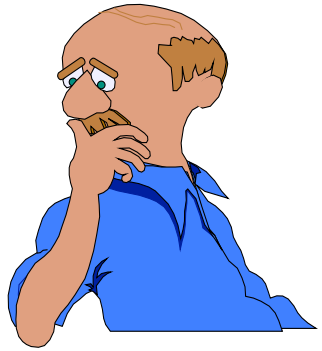
Le langage de définition et de manipulation des données

- **Définition et manipulation peuvent être faites dans n'importe quel langage mais il existe des langages spécifiques en bases de données : LDD**



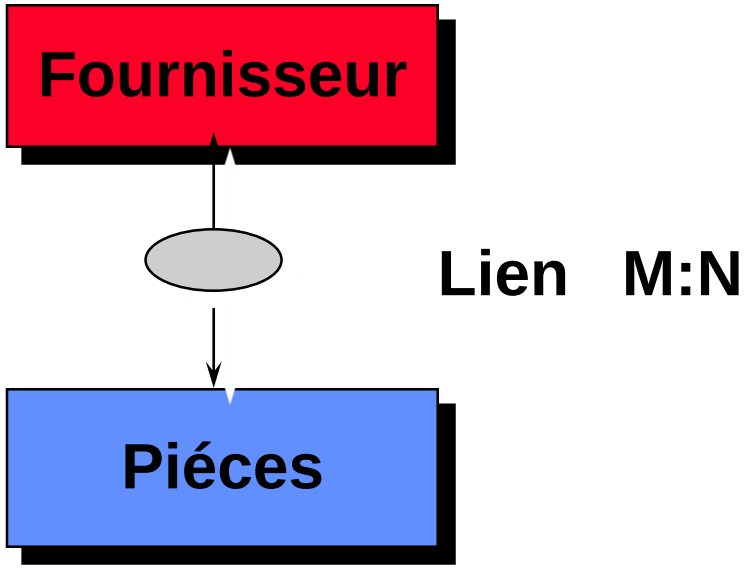
Exemples de requêtes

- **Q1 : Trouver toutes les sessions de Montréal**
- **GU COURS**
- **faire tant qu'il existe une session**
- **GN SESSION(VILLE= MONTREAL)**
- **fin**
- **Q2 : Trouver toutes les sessions du cours 3030**
- **GU COURS(#COURS=3030)**
- **Faire tant qu'il existe une session**
- **GNP SESSION**
- **GN, GU fixent la parenté --->position courante**
- **Que donnent les requêtes suivantes ?**
- **GU COURS**
- **GN**
- **GN GN GN**

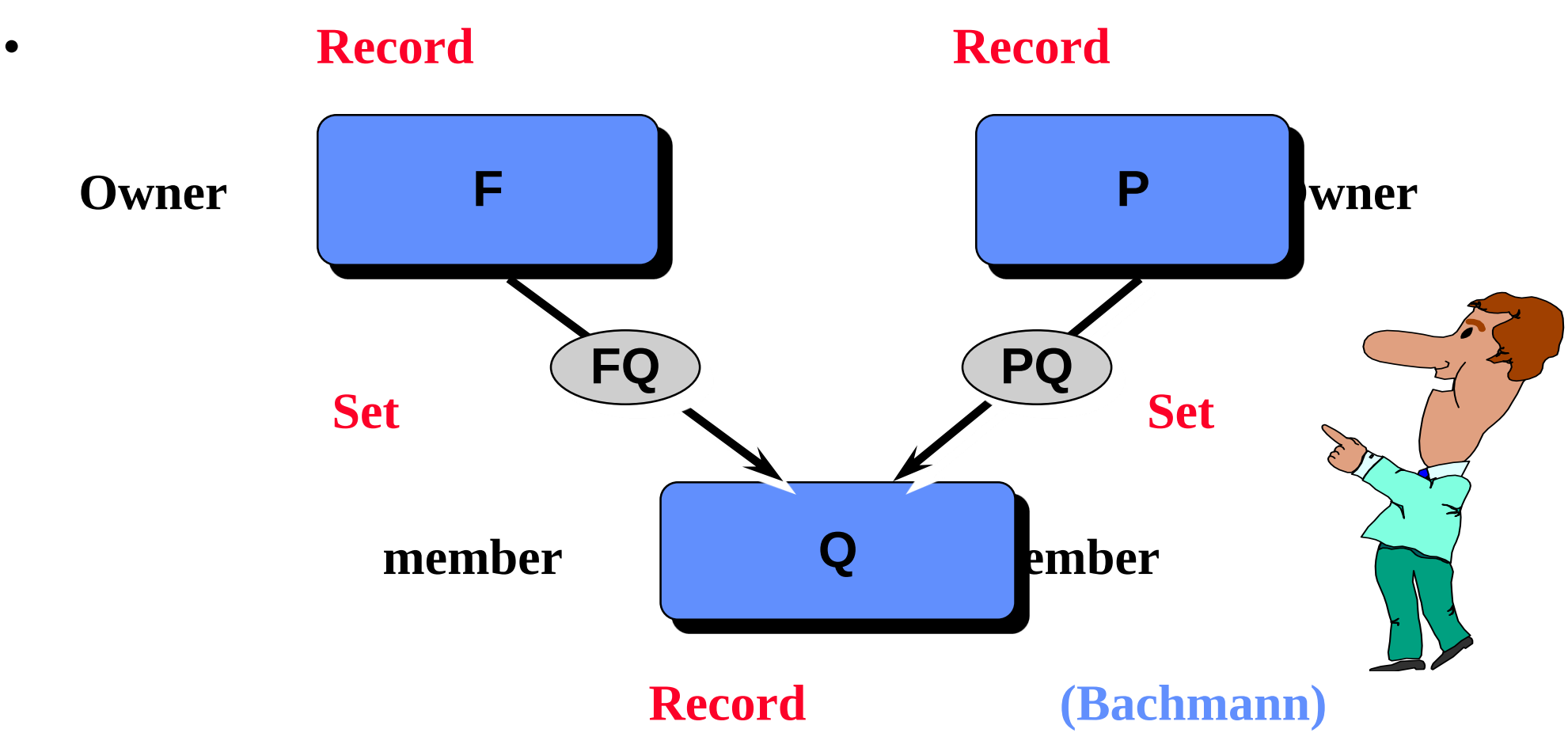


Le modèle et langage réseau

- **Network model**
- **Proposé par le DBTG du CODASYL, pour améliorer le modèle**



- **relation plusieurs-plusieurs**
- **Un lien est une connexion entre un fournisseur et une pièce. Pour un fournisseur tous les liens sont placés sur une chaine partant (du) et retournant au fournisseur**

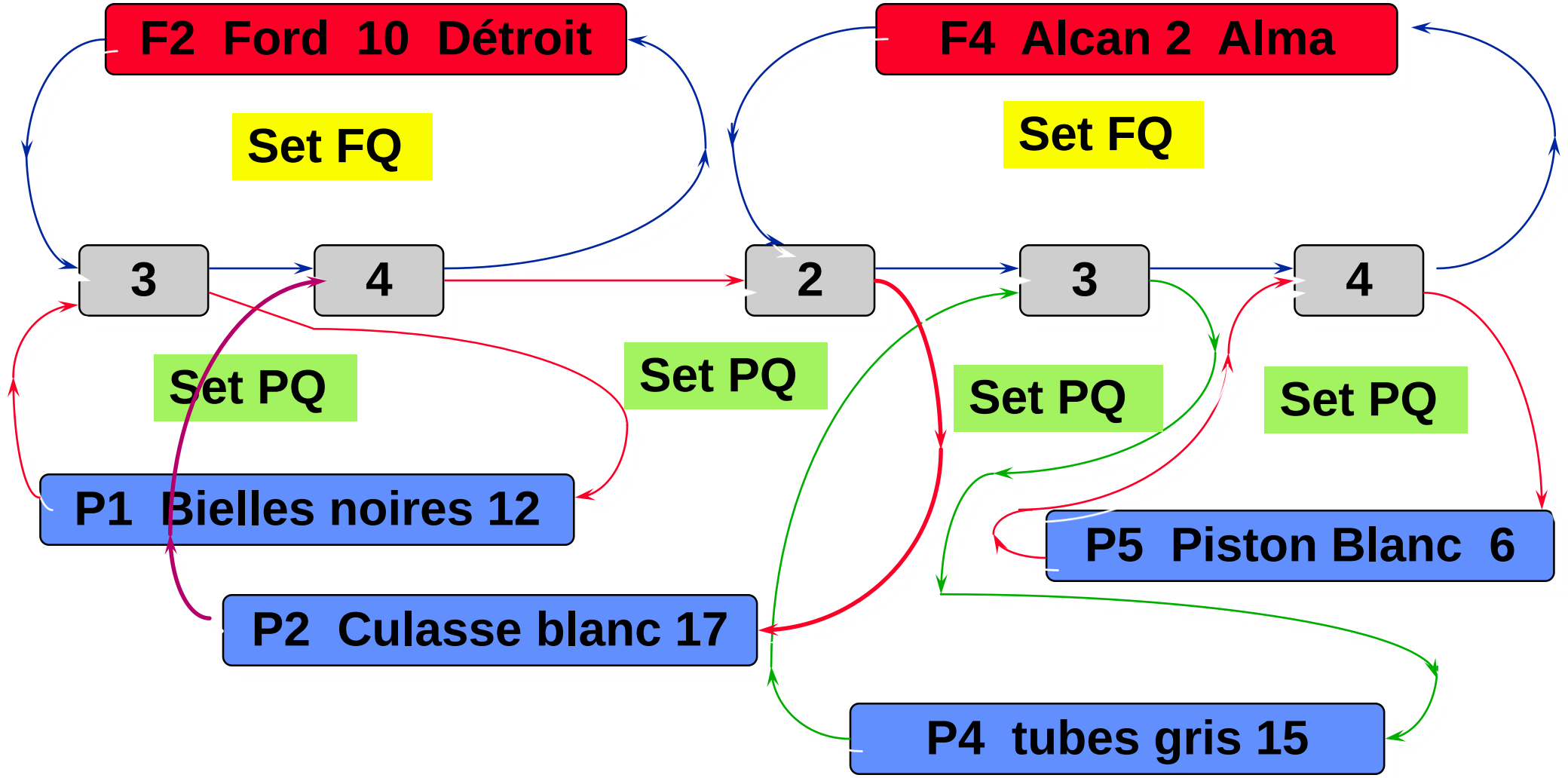


Un seul propriétaire de Set

0 ou N membres

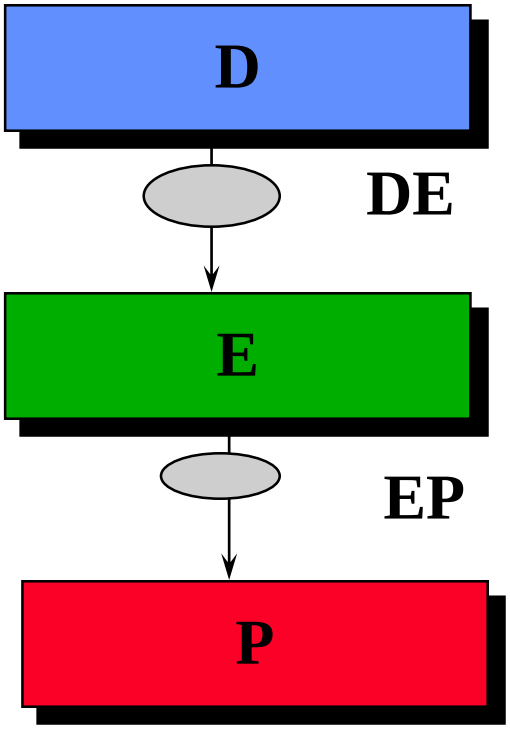
Mais → un record peut être

- **membre de plusieurs sets**
- **ou membre et propriétaire de plusieurs sets**
- **ou propriétaire de plusieurs sets**


F2 Ford 10 Détroit
Set FQ
3
4
Set PQ
Set PQ
P1 Bielles noires 12
P2 Culasse blanc 17
F4 Alcan 2 Alma
Set FQ
2
3
4
Set PQ
Set PQ
P5 Piston Blanc 6
P4 tubes gris 15

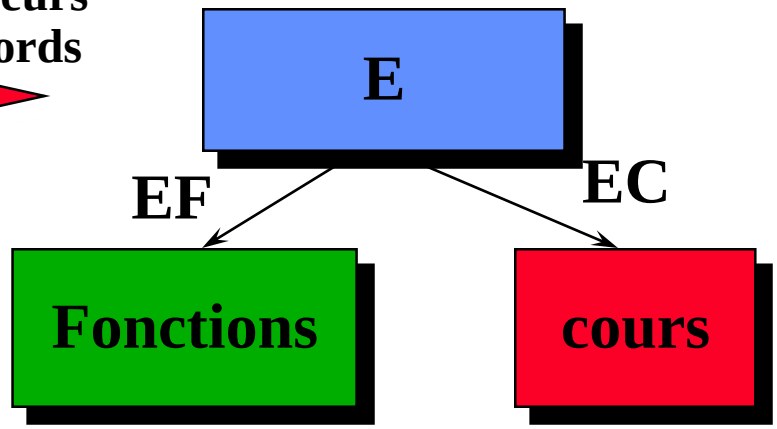
- **Résoud certains problèmes du modèle hiérarchique**
  - **Ajout : il est facile d'ajouter une nouvelle pièce (sans liens)**
  - **Suppression : on peut supprimer F2 sans perdre l'information sur la pièce P2**
  - **Mise-à-jour : on peut modifier (changer la couleur d'une pièce par exemple) sans amener une incohérence car la pièce existe une seule fois**
- **Inconvénients**
  - **proche de la structure de stockage**
  - **programmation peut-être rapidement complex**
  - → **Et si P1 est maintenant fourni par F4 ???**

- **Systèmes : IDS, IDMS**

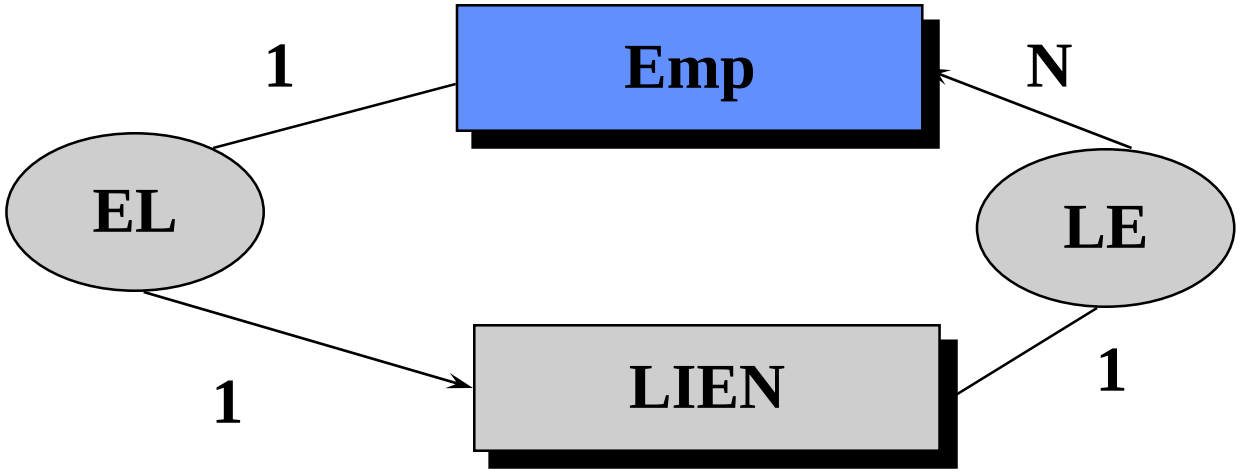


Hiérarchie linéaire à plusieurs niveaux

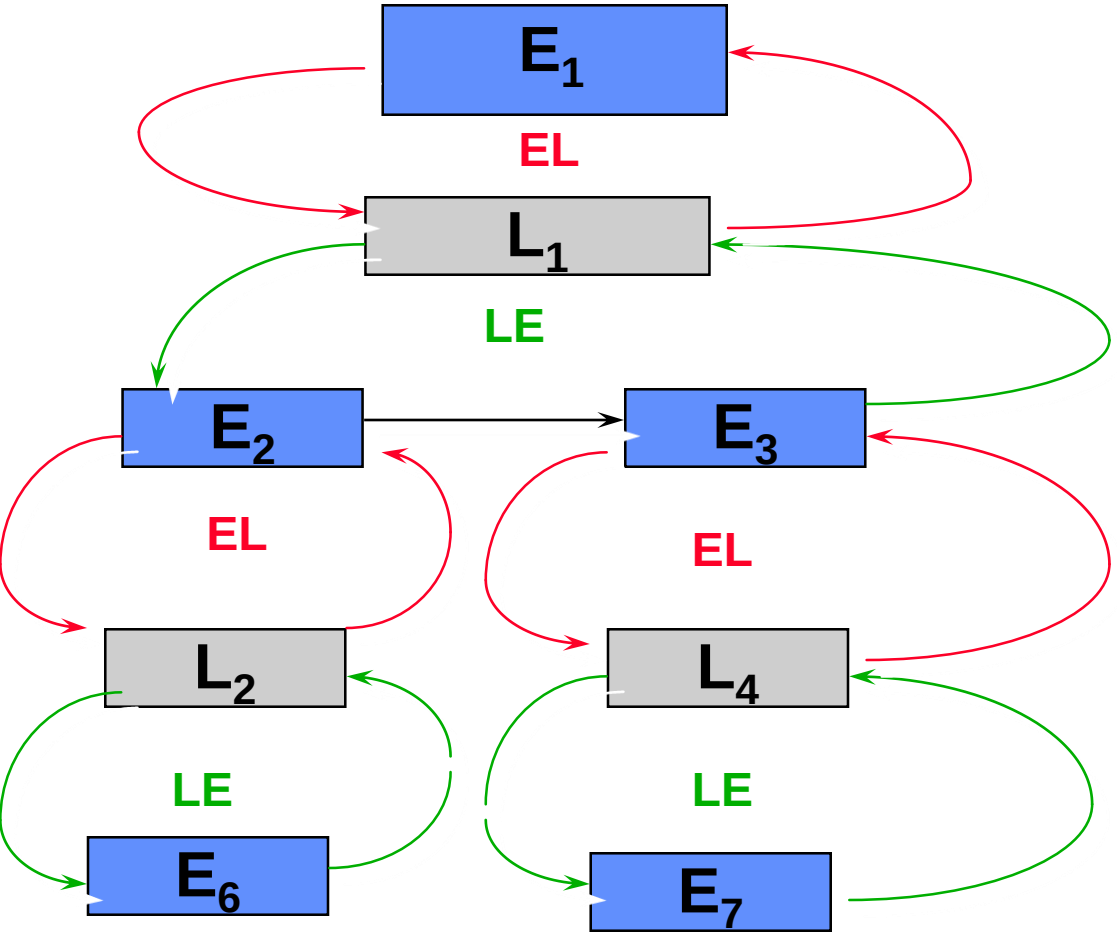
H avec plusieurs types de records



Différents modèles réseau...

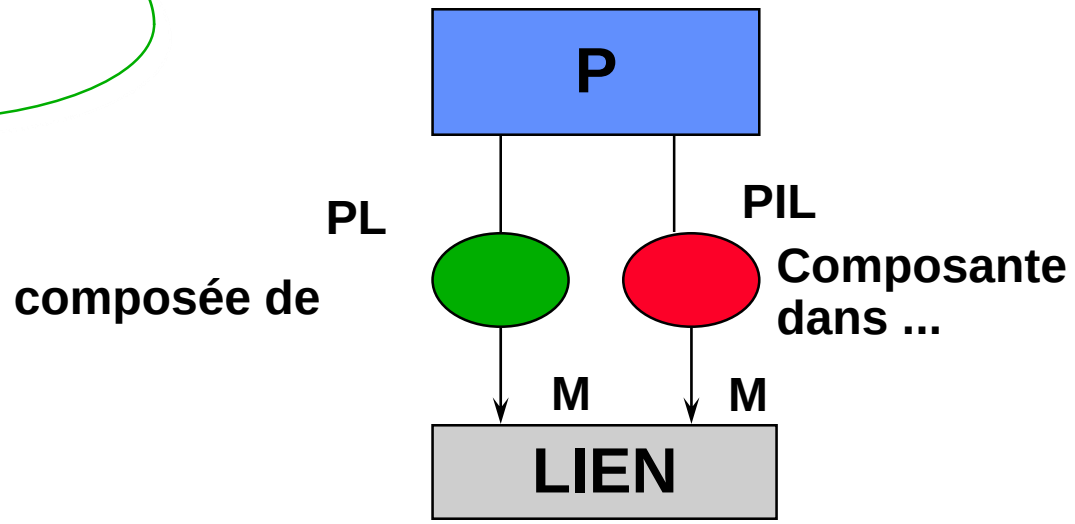


H avec même types de records à plusieurs niveaux

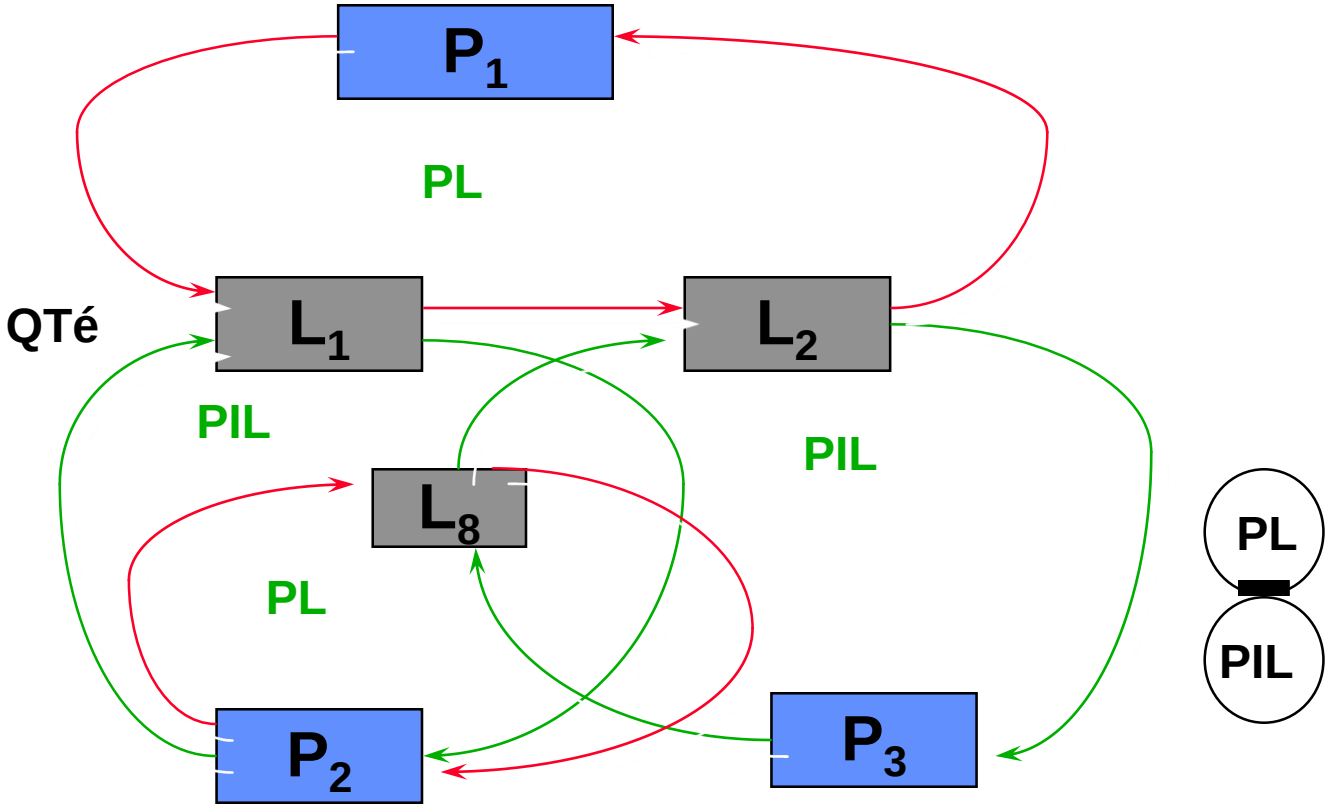


Occurrences employés-employés

Réseau à un seul type de record



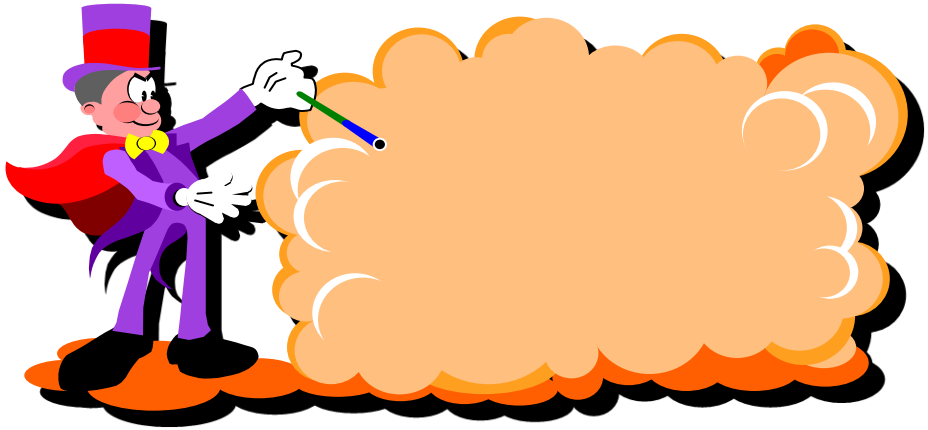
- **Exemple de réseau à un seul type de record**

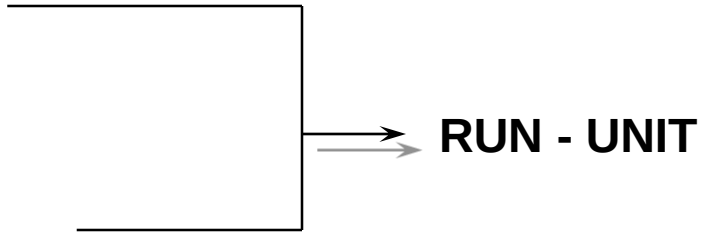


-> "P1 est composée de L1 pièces P2 et L2 pièces P3", ...

Les principales instructions

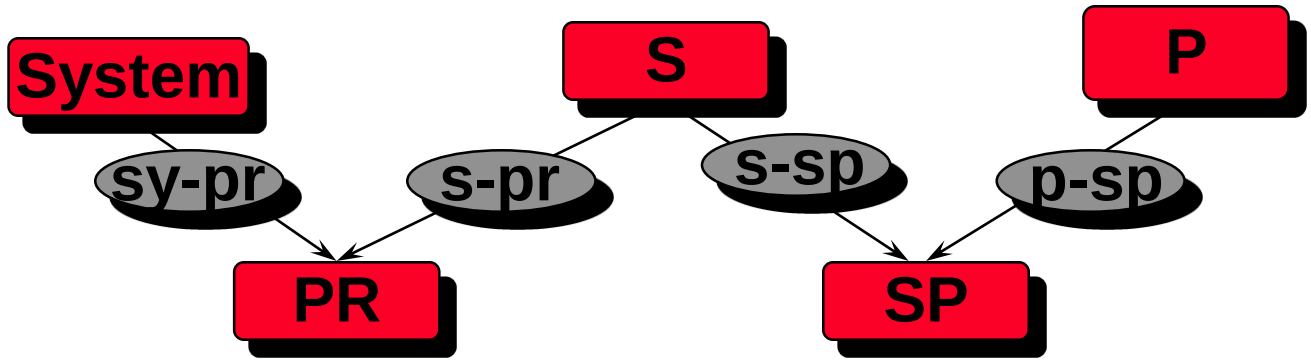
- FIND
- GET
- ERASE
- STORE
- MODIFY
- CONNECT
- DISCONNECT
- RECONNECT





a) **FIND**:

- all
- current
- <nb> within
- owner



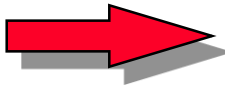
- **Find all** **<listp> <nomrecord>**

  listp : liste où seront mis les pointeurs des records retrouvés

  [within] <nom de set>

  [using] <liste d'item>

  where <condition>



Trouver toutes les pièces de couleur rouge

Find all listp1 P using couleur

> couleur: rouge

listp1 contient une liste de pointeurs des occurrences répondant au critère

b) **Find current within** <nom record>

<nom set>

Find SP using Pno

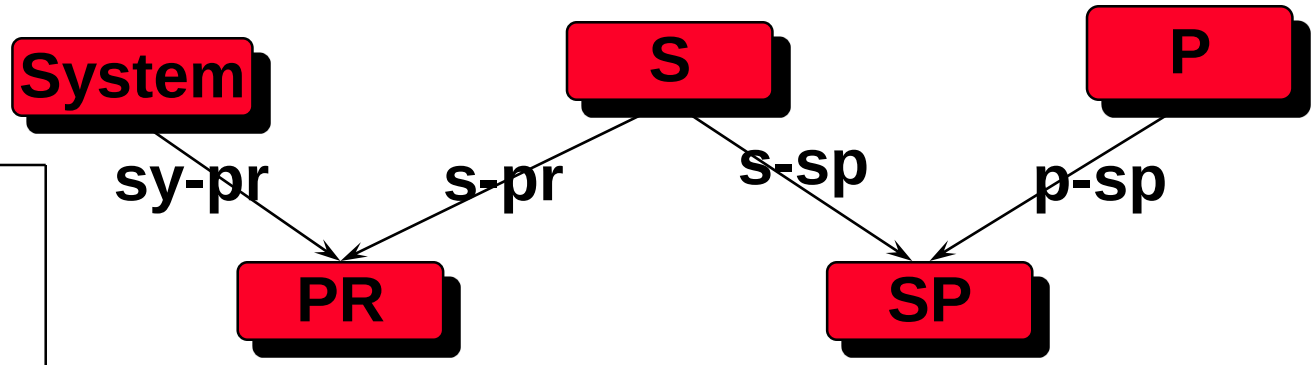
Pno = 1400

Find owner within S-SP

Find current within S-Pr

c) **Find**

- First
- Last
- Next
- Prior
- <entier, relatif>



- Within <nom set> [using] <liste d'item>

  where <condition> [For update]

Trouver la 5ème pièce qui est un boulon

Find 5 P where Pname = 'Boulon'

where Pname matches 'vis *'

vis à métaux

vis à bois

Where

- **LT**
- **LE**
- **EQ**
- **NE**
- **GT**
- **GE**
- **MATCHES**
- **CONTAINS**

(+ opérateur)

AND

OR

NOT

Trouver les pièces qui contiennent du bois

Find P where Pname contains 'Bois' (pas de différences avec minuscules)

vis à bois

scie à bois

allumettes en bois

- **Find last S where Sname contains 'al'**

d) Find owner within <nom set> (retrouve le propriétaire d'un set)

Trouver le fournisseur qui fournit un projet dont le budget est supérieur à 10 000 et le nom est delta

Find 1 PR using Nom, Budget

>nom: delta

>Budget: 10 000

- **Find owner within S-PR**

e) Free all [from <liste>]

current

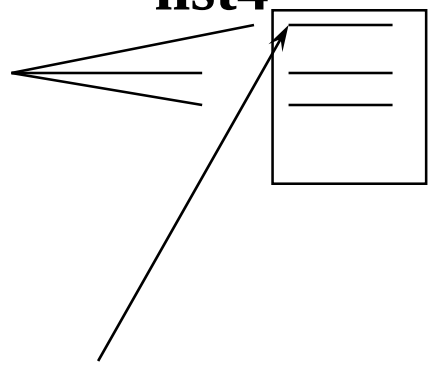
Efface les entrées (donc les adresses) contenues dans une liste

Free all From List1

- **Free all : toutes les listes**
- **Free current within <nom record>**
- **<nom set>**
- **Met l'indicateur courant à 0**

- Free current within P-SP
- Free <entier> within <list>
- <first>
- <last>
- Find all list4 S using ville list4

  >ville : Montréal



- Get S
- Free first within list4

f) Fetch (même syntaxe que le Find)

- transfert des données

> Find first SP where Pno EQ 1500

> Find owner within S - SP

> Fetch current within S - PR

g) GET <nom Record>

- **<liste d'items> of <Record>**

  > GET S

  > GET nom, ville of S

h) Keep current

- **Keep current [within] <nom record>**
  <nom set> [using] <liste d'item>

> Fetch first P within P - SP using Pno

> Pno : 1500

> Keep CURRENT within P - SP using list.fournies

i) Store <nom record> : ajoute un nouvel enregistrement

j) Erase [all] <nom record>

k) Connect

- **place un enregistrement courant dans un ou plusieurs set, l'occurrence de ce set étant déterminée par la position courante et la clause Order**
- **Connect <nom record> to**
  [all] si record défini comme membre des sets
  <liste de SET>

l) Disconnect <nom record> from
[all]
<liste de SETS>

m) Reconnect <nom record> within [all]
<liste de SET>

n) Retention

Le modèle et langage relationnel

- **Basé sur la théorie des Relations**
- **Étant donnés les ensembles D_1, D_2, ... D_n** **Codd(IBM)**

  R est une relation sur ces n ensembles si c'est un ensemble ordonné de n-uplets <d1, d2, ... dn>
  tels que d1 ε D1, d2 ε D2, ... dn ε Dn

- **D_1, D_2, ... D_n sont des domaines de R**
- **n est le degré de la Relation**

-> Ex: Relation Pièce

Pièce

{ tuple, n-uplet, ruple

domaines/ **Attributs**

(Pno,	Pnom,	couleur,	poids)
P1	boulon	blanc	55
P2	vis	noir	2
P3	bouchon	rouge	7
P4	clé	vert	3
P5	volant	jaune	15

- **Il n'existe pas 2 n-uplets identiques**
- **L'ordre des colonnes n'importe pas**
- **L'ordre des lignes n'importe pas**
- **Un domaine particulier est la "clé primaire" (Ex.: Pno)**
- **Chaque valeur de Pno est différente**

* **Chaque valeur identifie de manière unique le n-uplet**

- **Cas où deux domaines(attributs) représentent le même ensemble (domaine):**

Exemple : relation **Composant (PnoM, Pnom, Qté.)**

Composant

Clé primaire

Pno, M	Pnom,	Qté.
P1	P2	2
P1	P4	4
P5	P3	1
P3	P6	3
P6	P7	9
P5	P6	8
P2	P4	4

Exemple d'un modèle relationel

FOURN

Pno	Pnom	cat	ville
F1	Ford	1	Détroit
F2	GE	2	N.Y.
F3	Hitachi	2	Tokyo
F4	Radio-S	4	Toronto
F5	Motorola	4	L.A.
F6	...	4	L.A.

P

Pno	Pnom	couleur	poids
P1	Culasse	blanc	25
P2	boulon	jaune	8
P3	Vis	blanc	5
P4	circuit	noir	2
P5	capot	beige	5
P6	transfo	noir	58

FP

CE

F1 P2

Fno	Pno	Qte
F1	P1	1 000
F1	P2	10 000
F1	P4	10 000
F2	P4	100
F2	P6	200
F3	P4	300
F4	P4	1 000
F5	P4	200

- **Chaque fournisseur fournit des pièces et chaque pièce est fournie par des fournisseurs (correspondance plusieurs-plusieurs)**
- **Aucune clé primaire ne peut être nulle (en tout ou partie)**
- **Intégrité référentielle Chaque valeur de #P dans FP doit être égale à cette #P dans P.**

clé candidate ≠ clé alternée

- **Ajout:** **Il est facile d'ajouter une nouvelle pièce (P7) -> ajout d'un nouveau tuple**
- **Suppression:** **On peut supprimer F1 sans perdre l'information sur la pièce fournie**
- **M.A.J.:** **On peut changer la couleur de la pièce P2 (en jaune) sans problèmes de recherche et sans possibilité d'incohérence.**

⇒ **Résoud de nombreux problèmes des modèles hiérarchiques et en réseau.**

Caractéristiques des SGBD relationnels

- **Suivant Codd, un système est relationnel<=> il supporte au moins les propriétés suivantes:**

  1- Il y a une Base de données relationnelle (la base de données est perçue par l'usager selon les tables)

  2- On a au moins les opérations:

  sélection, projection, jointure naturelle

  sans détails sur les accès physiques entraînés par ces opérations.

Par exemple, si on ne peut que sélectionner des enregistrements indexés (champs indexé), ce n'est pas un Système Relationnel (S.R.).

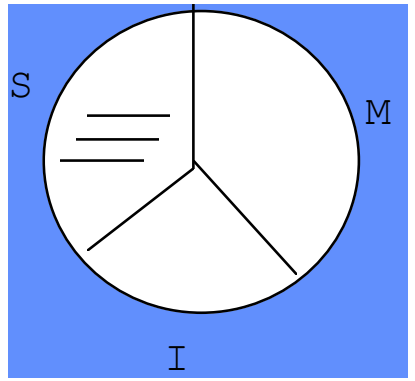
RAISONS

- **S, P, J sont un sous-ensemble de l'algébre (très utile). Ils résolvent presque tous les problèmes.**
- **un système qui supporte uniquement l'aspect relationnel (tables) ne permet pas la qualité d'utilisation d'un S.R.véritable**
- **un système qui fournit les opérateurs relationnels mais nécessite une prédéfinition des chemins d'accès n'obéit pas au principe d'indépendance physique.**
- **les opérateurs relationnels doivent être implémentés avec optimisation => tâche non triviale => raison du délai d'apparition des SGBD industriels.**

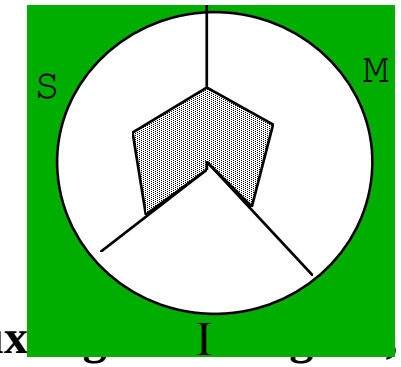
Schématiquement

- Chaque secteur => une propriété fondamentale du modèle (structurale, intégrité, manipulation) (S I M)
- Le degré de la propriété
  - » Systèmes tabulaires (sans set d'opérateurs)

    ce ne sont pas des relationnels.



  - » tables + opérations de S, P, J (pas d'autres)
    est relationnel **minimal**
  - » tables + tous les opérateurs
    ==> **Relationnel complet**



⇒ DB2, SQL/DS, INGRES, ORACLE RdB/VMS, ...

  - » tous les aspects du modèle => notions de domaines et les deux
    est totalement relationnel

    actuellement aucun, mais en préparation.