De la technologie bases de données à la technologie Web (Tutoriel)

Nacer Boudjlida

INFORSID

Informatique des organisations
et systèmes d'information et de décision

De la technologie BD à la technologie Web (Nacer Boudjlida, LORIA)

Tutoriel associé à Inforsid
Nancy, 3 juin 2003

De la technologie bases de données à la technologie Web

Nacer Boudjlida

LORIA, UHP Nancy 1

http://www.loria.fr/~nacer

(nancy, Juin 2003)

Objectifs du tutoriel

Citation de VLDB (*http://www.vldb.org/future.html)*, Dec. 1999:

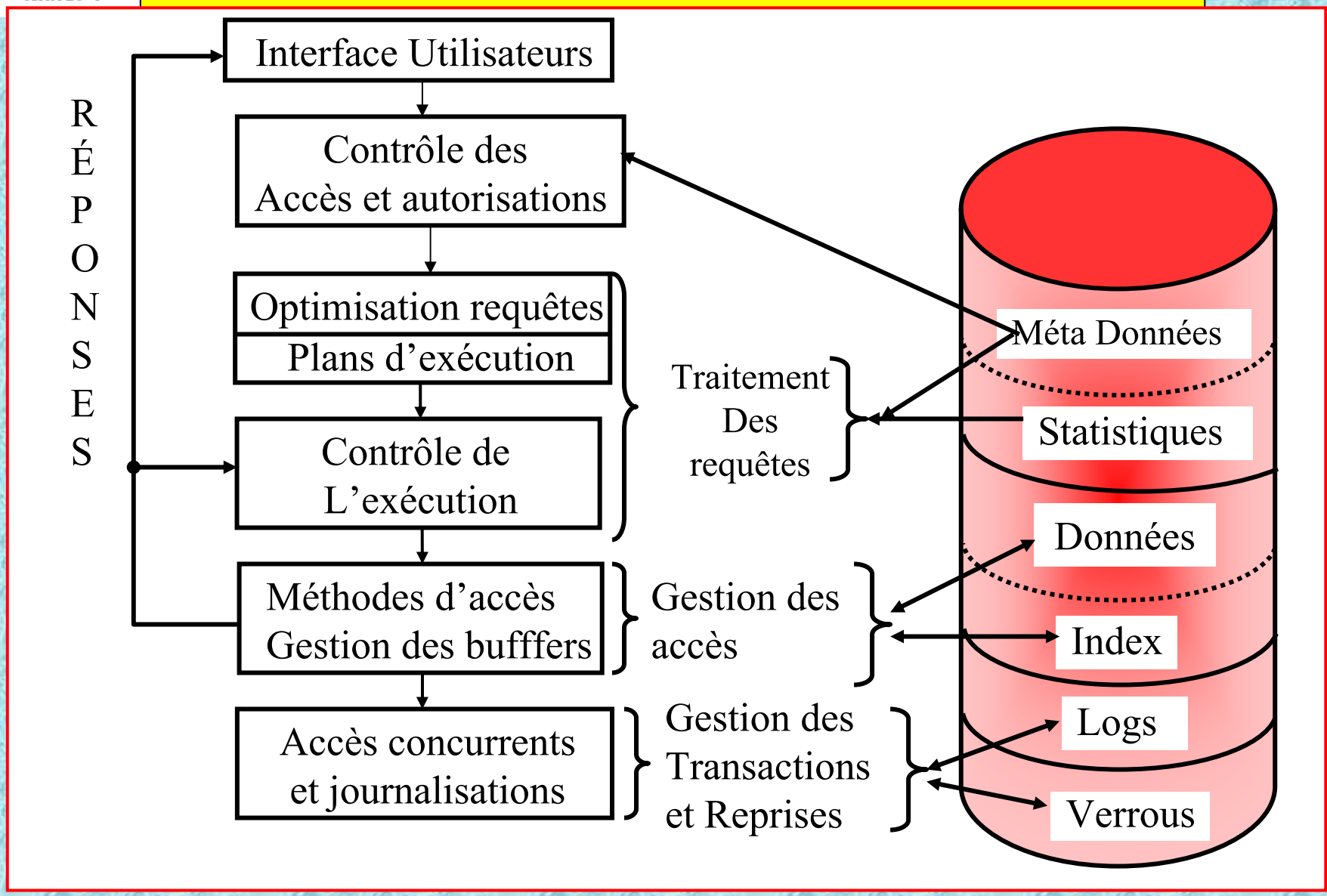
"… the database role of providing a storage manager, i.e. **core DB technology, remains central**… The **focus** of our community should turn to the investigation of **how core technology can become more widespread and usable**, by concentrating on the description of **new application areas**, on the method and tools for data analysis, design and integration, on the **technologies for data deployment in modern architectures** (middleware, wireless technology, the Web), and in general on all problems and **challenges** which are due to the need of **using very large databases in new contexts**."

Fonctions d'un SGBD

1. *Description* ("objets", attributs, relations)
2. *Manipulation* : Langages déclaratifs (orientés ensembles)
3. *Confidentialité et Intégrité*
4. *Accès concurrents*
5. *Sécurité de fonctionnement*

- *Efficacité* : Accès et Traitement des Requêtes

Architecture d'un SGBD



©Nacer.Boudjlida@loria.fr

Classes de SGBD

- Systèmes et *modèles de représentation de données*
  - CODASYL (SGBD hiérarchiques/réseaux)
  - Relationnel (SGBD R)
  - Orienté Objets (SGBDO)
  - Données Non ou Semi-Structurées *(Web-SGDB ?)*
- Evolution du domaine :
  - % autres domaines (Programmation, Réseaux, IA, etc.)
  - % besoins des nouvelles applications (CAD/CAM, CSCW, Wflow, Ingénierie du Logiciel, etc.)

Objectif du tutoriel

- *Limitation* à représentation et manipulation
  - Données (fortement) structurées
  - Objets et données non ou semi-structurés
- Similitudes et différences (SGBD-Web)
- *Evaluer* l'adéquation (d'une partie) de la technologie BdD par rapport à
  - nouveaux types d'applications
  - nouvelles architectures (Client/Serveur + Web)
- Version longue : *http://www.loria.fr/~nacer*

Structure de la Présentation

Partie I : *Survol* de la technologie relationnelle

- I.1- Modèle et Langages
- I.2- Traitement des requêtes (Centralisées, réparties)

Partie II : Objets complexes structurés

Partie III : Web et Bases de données

- III.1- Objets semi-structurés
- III.2- Web et Bases de données

I.1- Modèle et Langages Relationnels

- Concepts du Modèle de Données
  - *Domaine* : Ensemble de valeurs
  - *Attribut* : Nom associé à un domaine
  - *Relation* : Sous-ensemble du produit cartésien des domaines
  - Vision logique d'une relation : Table
    - *Colonnes* : Valeurs des attributs
    - Lignes : tuples, "faits''

- Déclaratifs, Orientés ensembles

1/2) Algèbre Relationnelle : Opérandes et résultats = Relations

- Sélection/Restriction **(σ)**, Projection **(Π)**
- Produit Cartésien **(X)**, Union **(∪)**, Différence **(\)**
- Note: Join(R, S, Cond) = σ_{Cond} (R x S)
  - Condition : R.Attribut θ S.Attribut
  - θ : Opérateur de Comparaison (=, <, >, etc.)

R ⋈ S
θ-expression

©Nacer.Boudjlida@loria.fr

2/2) SQL: *Dialecte Standard* des SGBDR

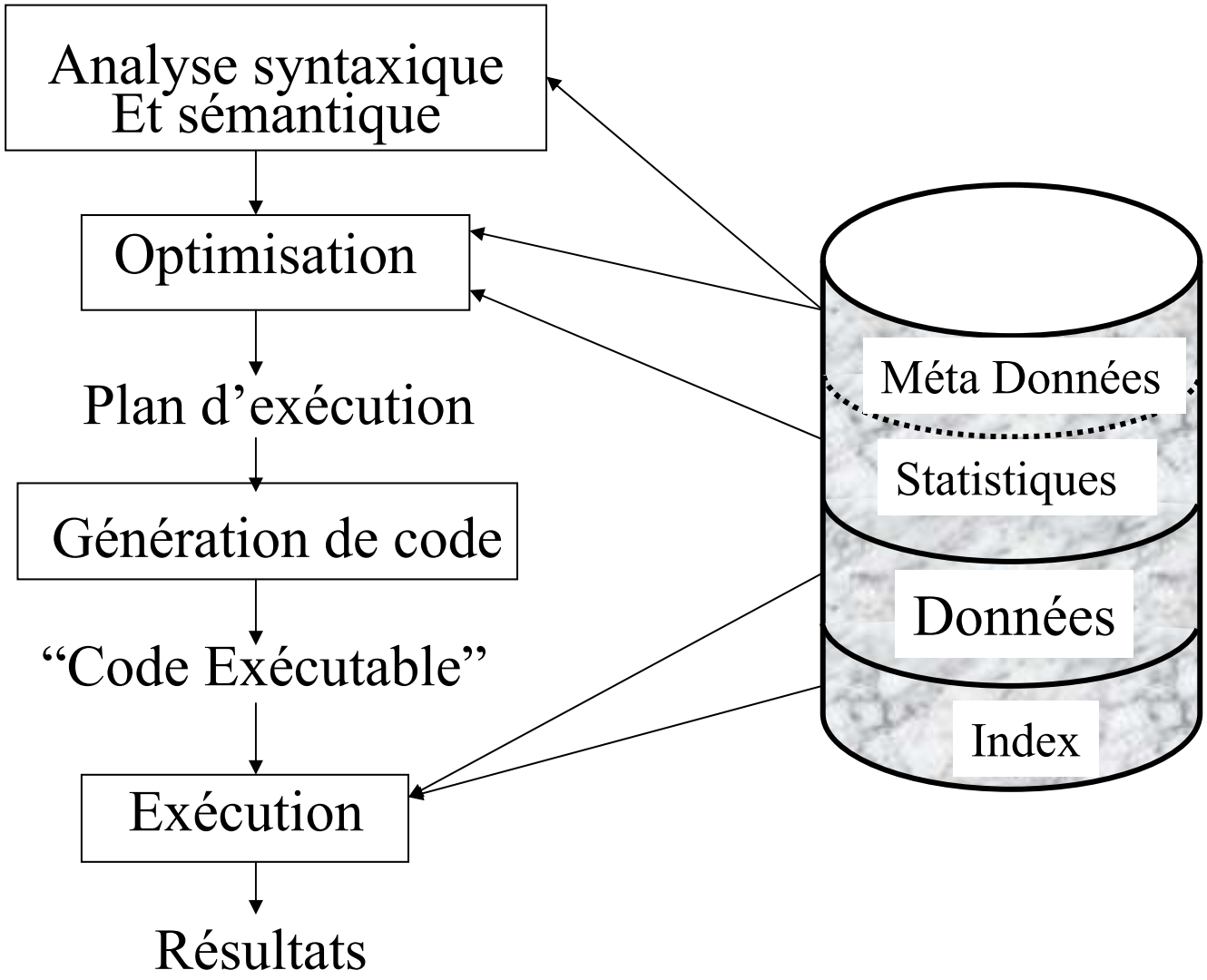
- **R → select * from R**
- **$\Pi_{(A1, ..., An)}$ (R) → select A1, …, An from R**
- **σ_C (R) → select * from R where C**
- **R x S → select * from R, S**
- **R ∪ S → select * from R union (select * from S)**
- **R \ S → select * from R minus (select * from S)**
- **Join$_{(\theta\text{-expression})}$ (R, S) = $\sigma_{(\theta\text{-expression})}$ (R x S)**

  → select * from R, S where θ-expression

Langages Relationnels (suite)

- Traitement de requêtes:
  - Minimiser le coût d'évaluation
  - Optimisation
    - Syntaxique (arbre relationnel, graphe de requête)
    - Sémantique
    - Estimation de coûts

Traitement des Requêtes (suite) : Le processus

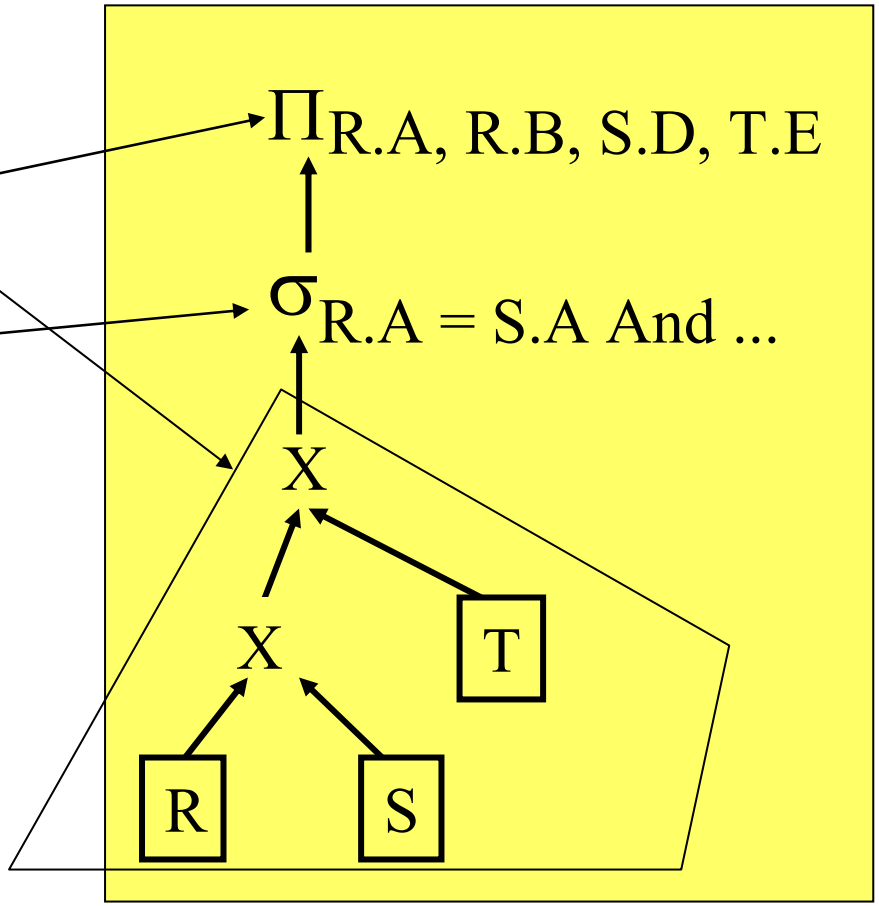


©Nacer.Boudjlida@loria.fr

- Requête SQL → Arbre algébrique
- Transformer l'arbre en utilisant
  - Propriétés des opérateurs algébriques (Associativité, Commutativité, Distributivité, etc.)
  - Heuristiques: Evaluer en 1er les opérations les moins coûteuses
    - « Faire descendre » les Sélections (↘ nbr de tuples)
    - « Faire descendre » les Projections (↘ nbr de colonnes)

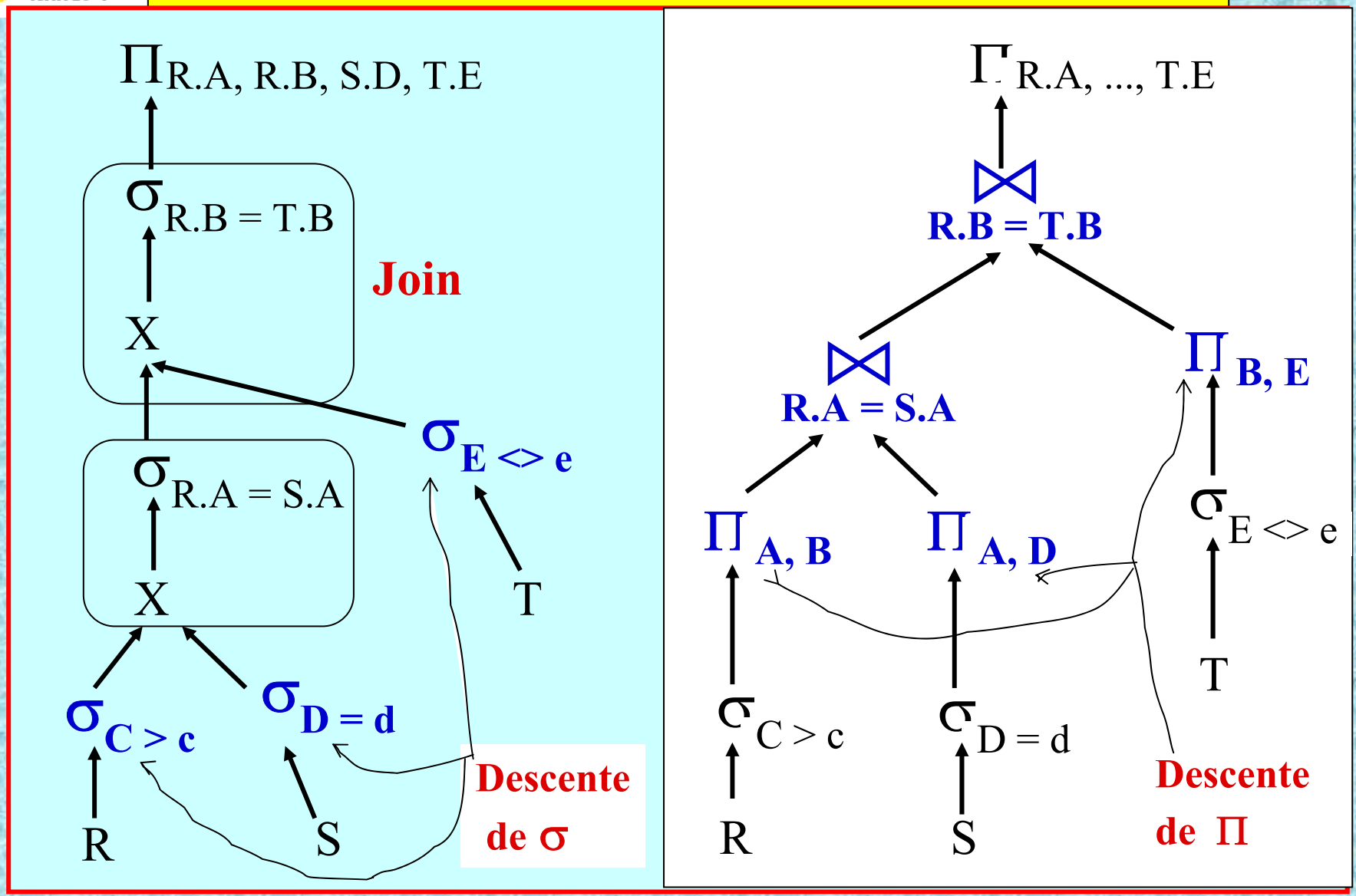
- Relations R(A, B, C, K), S(A, D, G), T(B, E, F)
- Requête

Select R.A, R.B, S.D, T.E
From R, S, T
Where R.A = S.A
And R.B = T. B
And C > c And D = d
And E <> e



©Nacer.Boudjlida@loria.fr

Traitement des Requêtes (suite) : Optimisation syntaxique



©Nacer.Boudjlida@loria.fr

Traitement des Requêtes (suite) : Bases distribuées

- *SGBD distribué* :

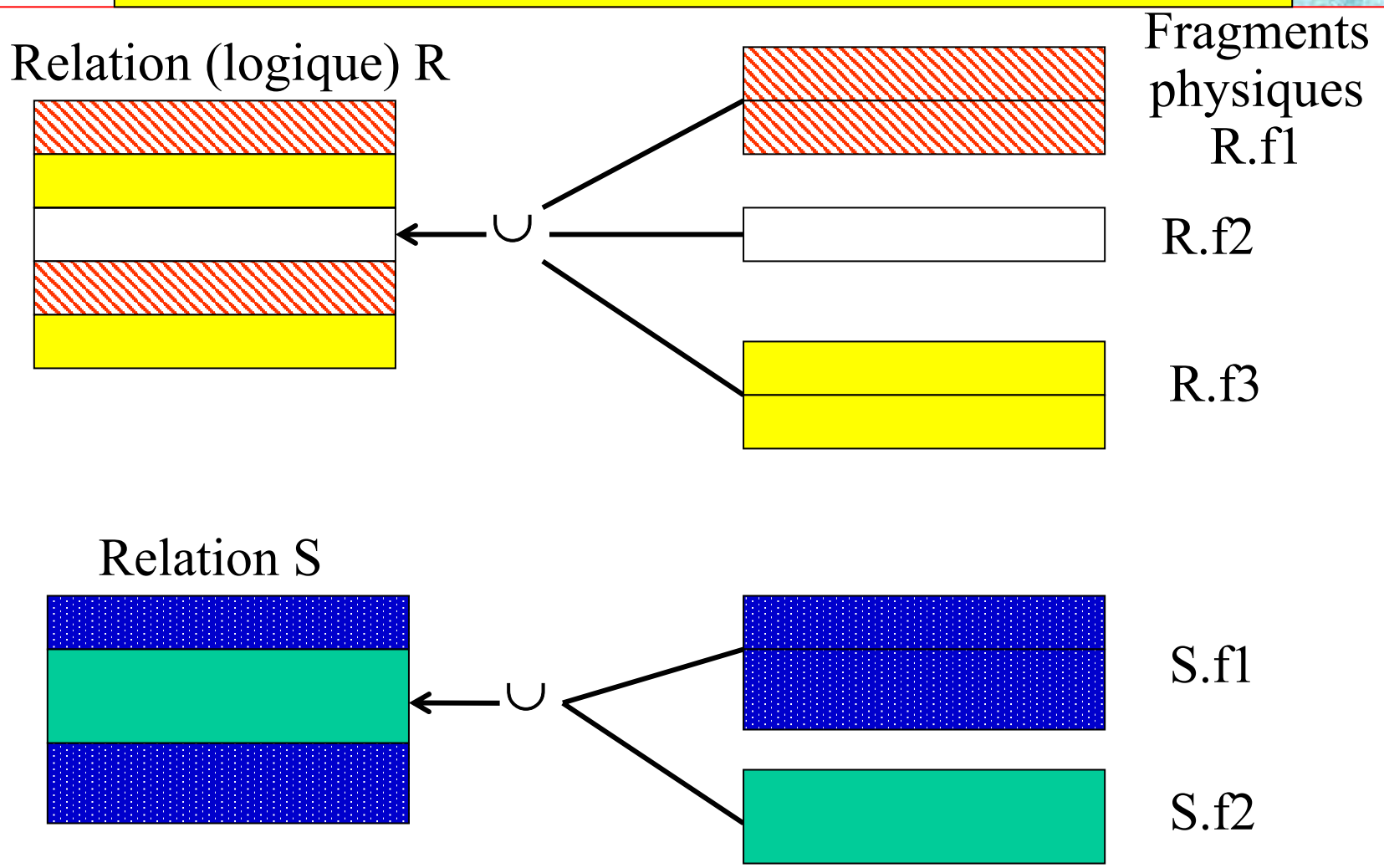
  SGBDs, ***autonomes***, éventuellement ***hétérogènes***, ***reliés par*** un réseau de communication et ***coopérants*** dans l'exécution de tâches.

- *Base distribuée*:

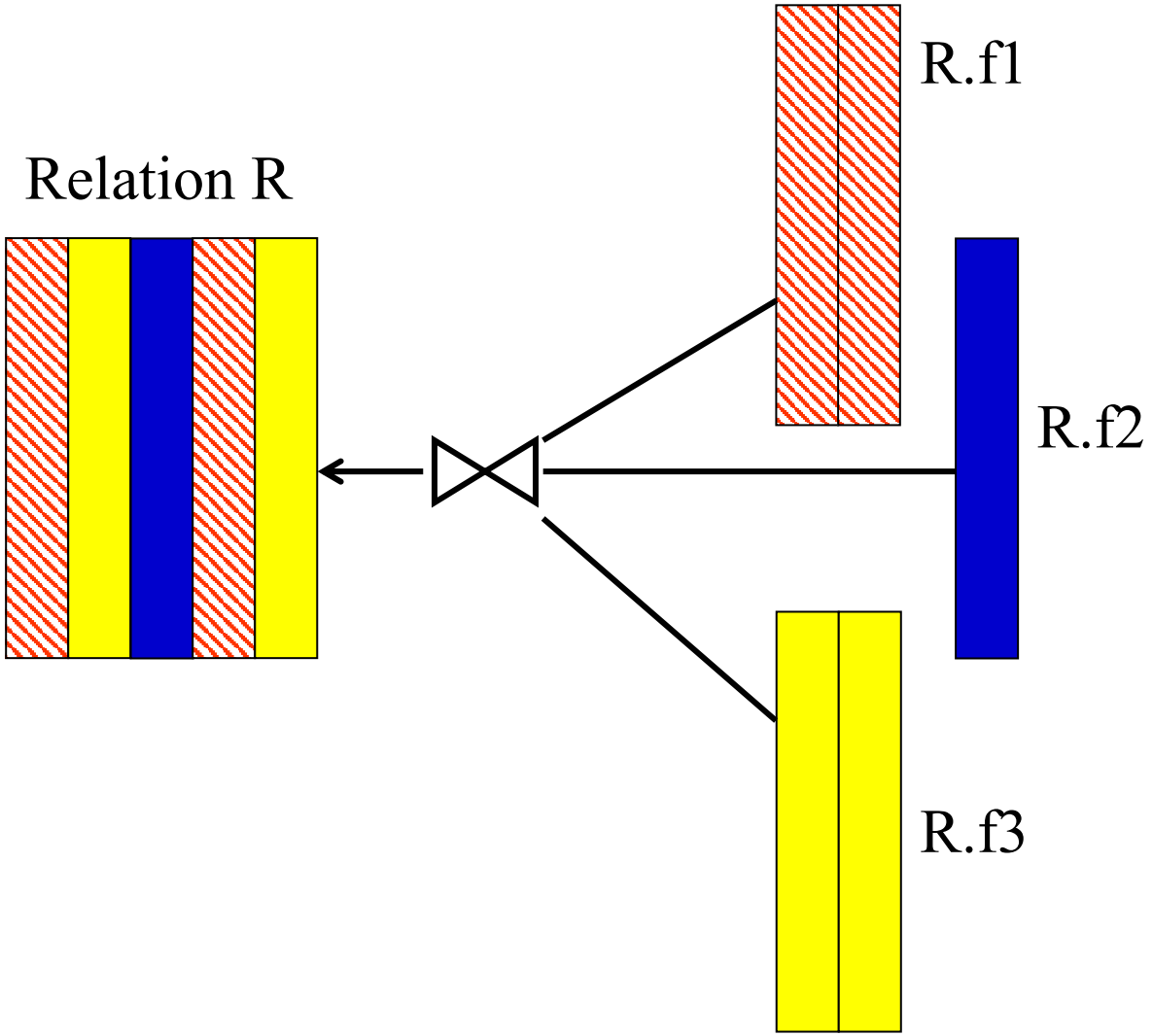
  Collection de bases de données ***logiquement reliées*** et ***physiquement distribuées sur un réseau***

©Nacer.Boudjlida@loria.fr

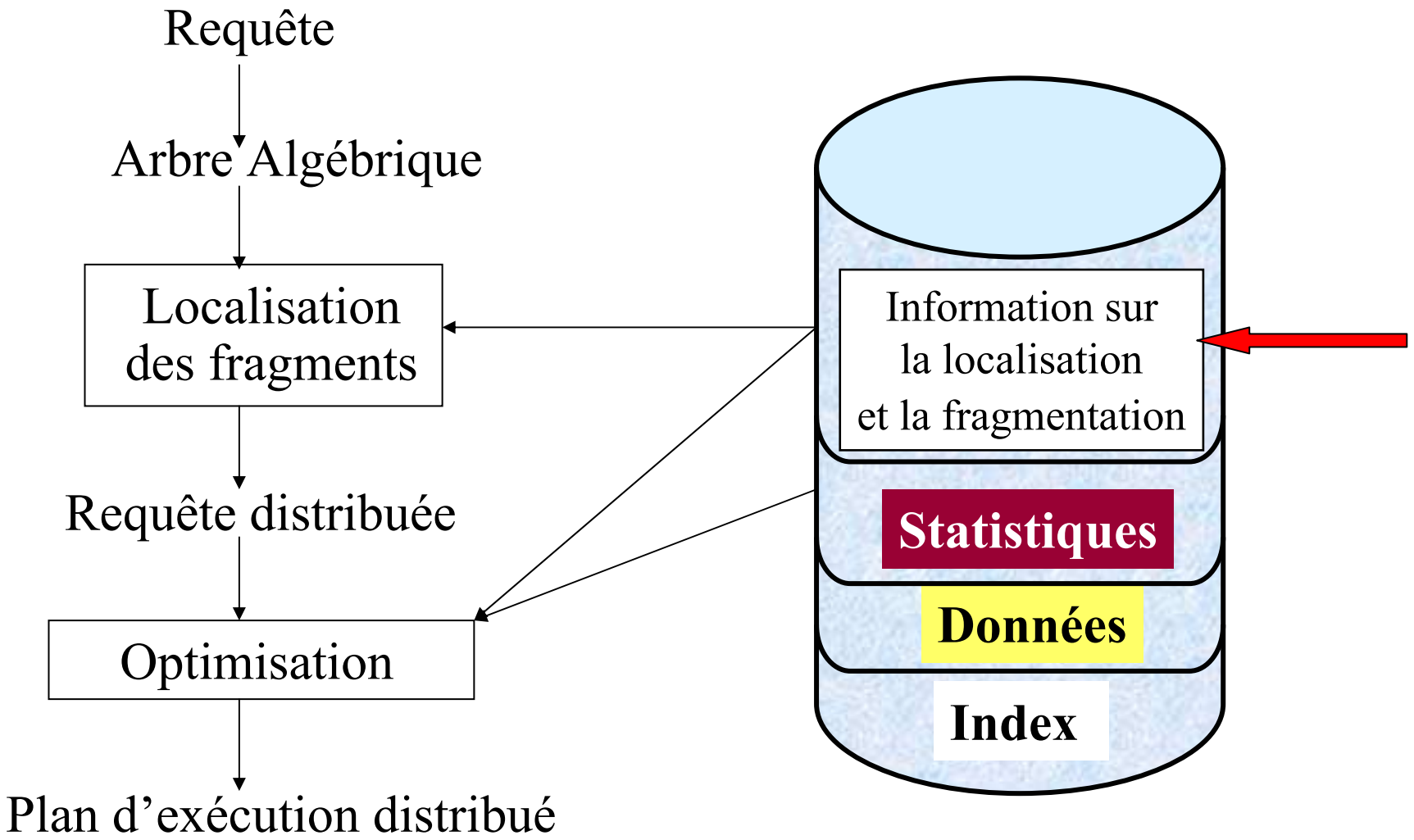
Bases distribuées : Fragmentation Horizontale



Bases distribuées : Fragmentation Verticale

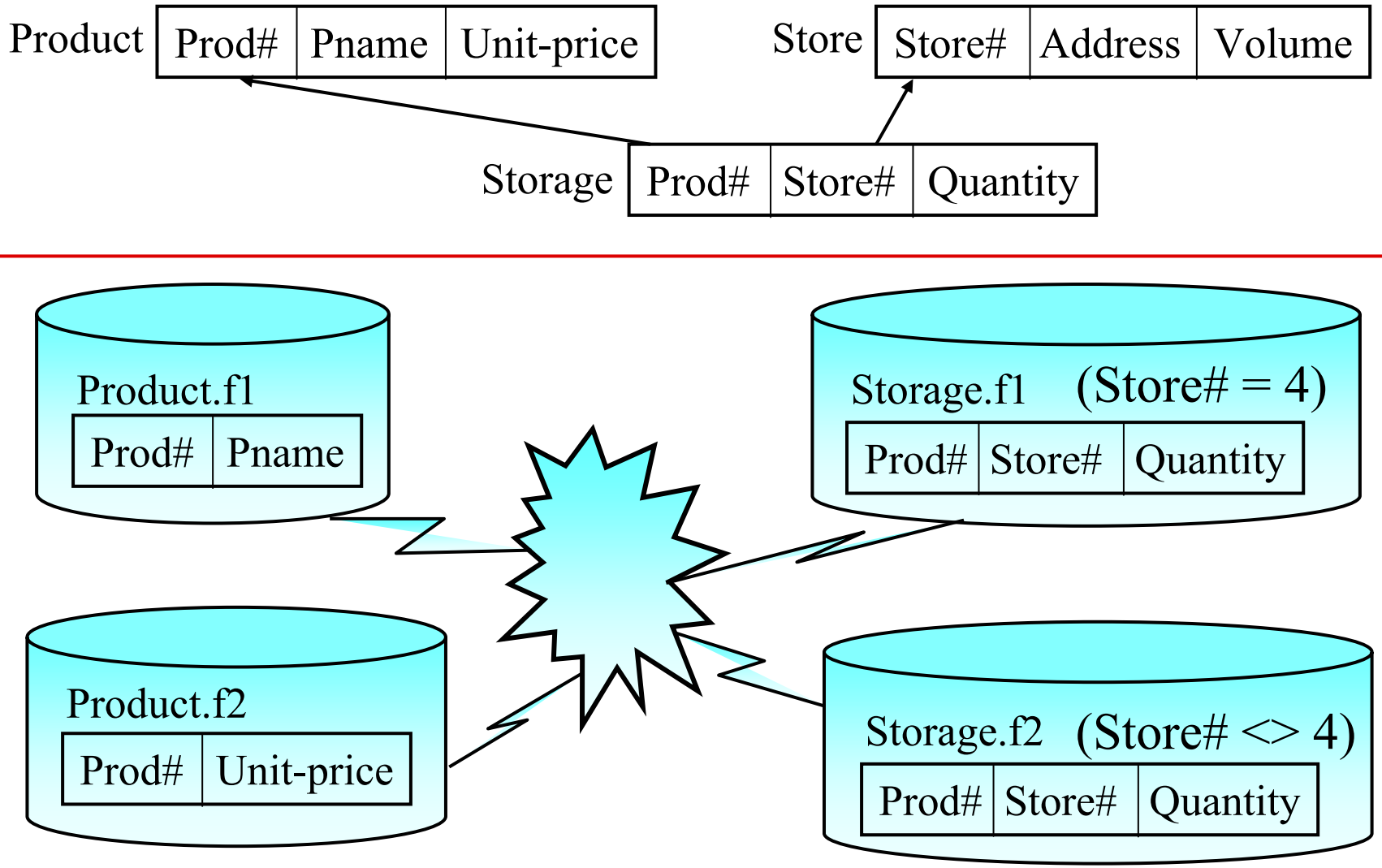


Bases distribuées : Traitement des Requêtes



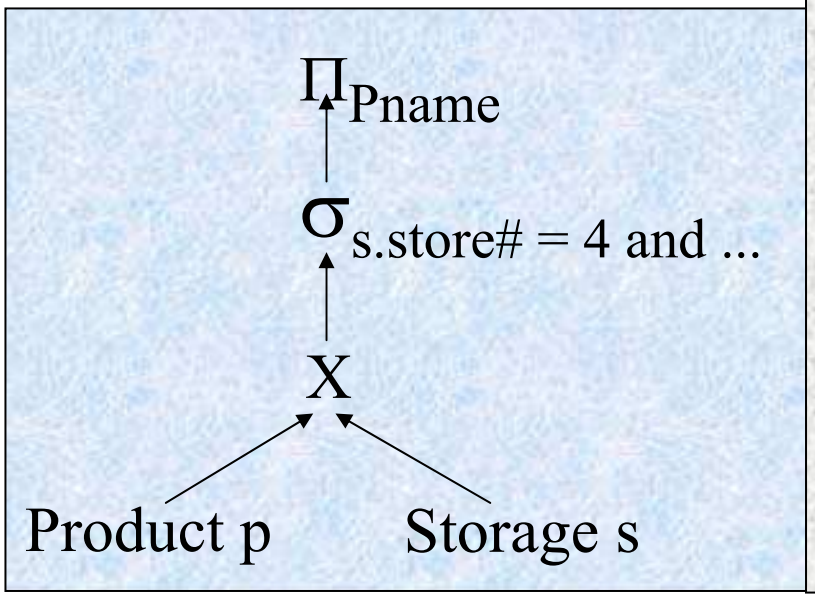
©Nacer.Boudjlida@loria.fr

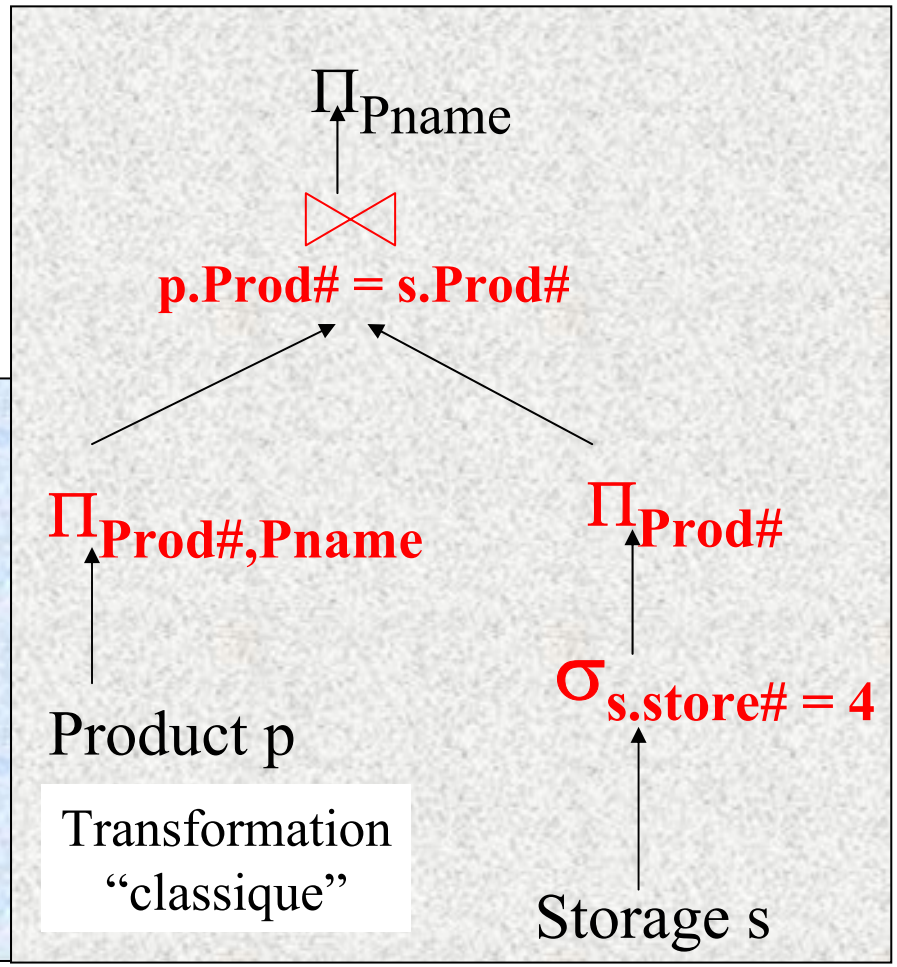
Bases distribuées : Traitement des Requêtes



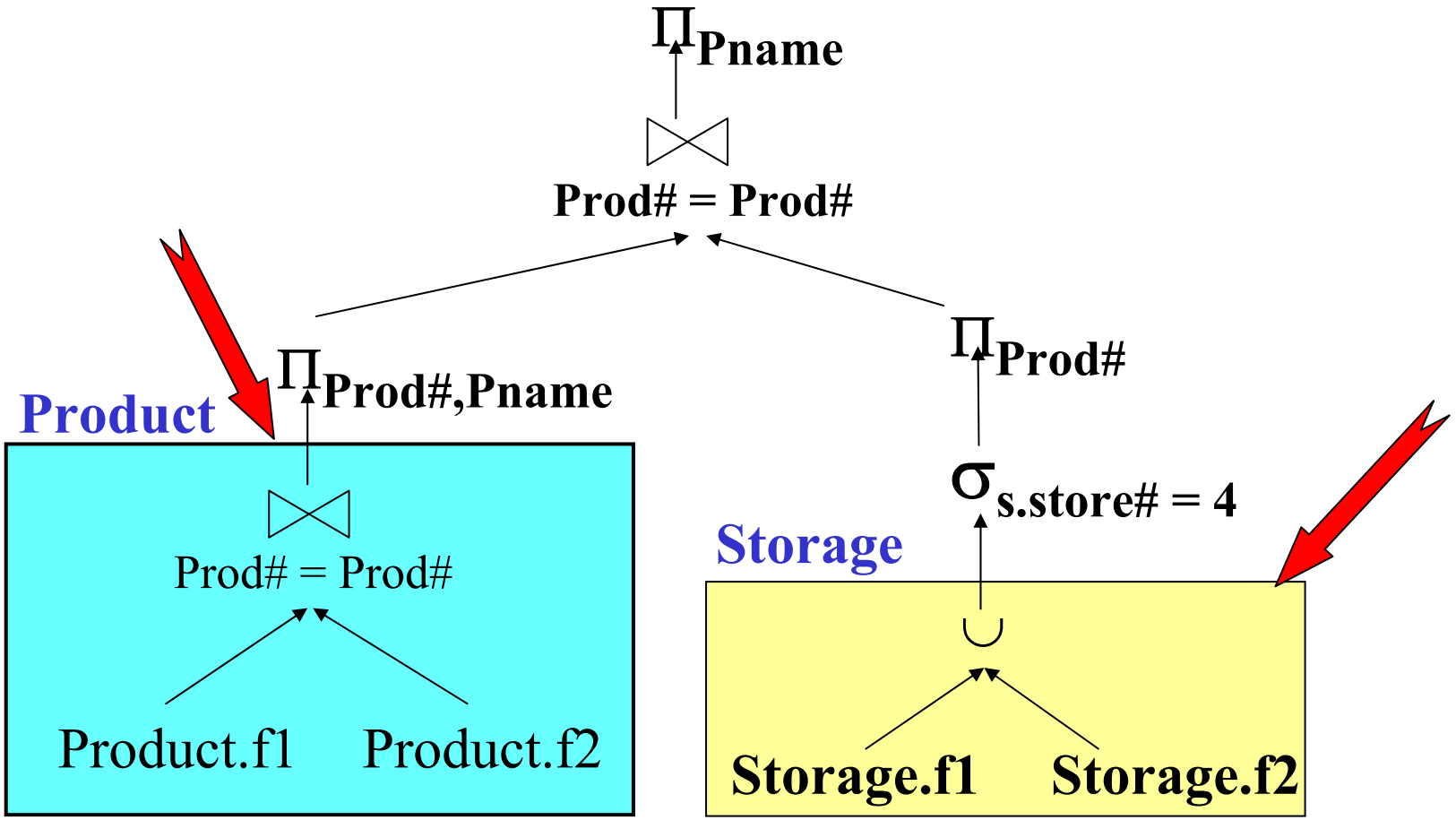
- Requête : Noms des produits du dépôt N° 4 ?

```
Select p.Pname
From Product p, Storage s
Where s.store# = 4
And     p.Prod# = s.Prod#
```

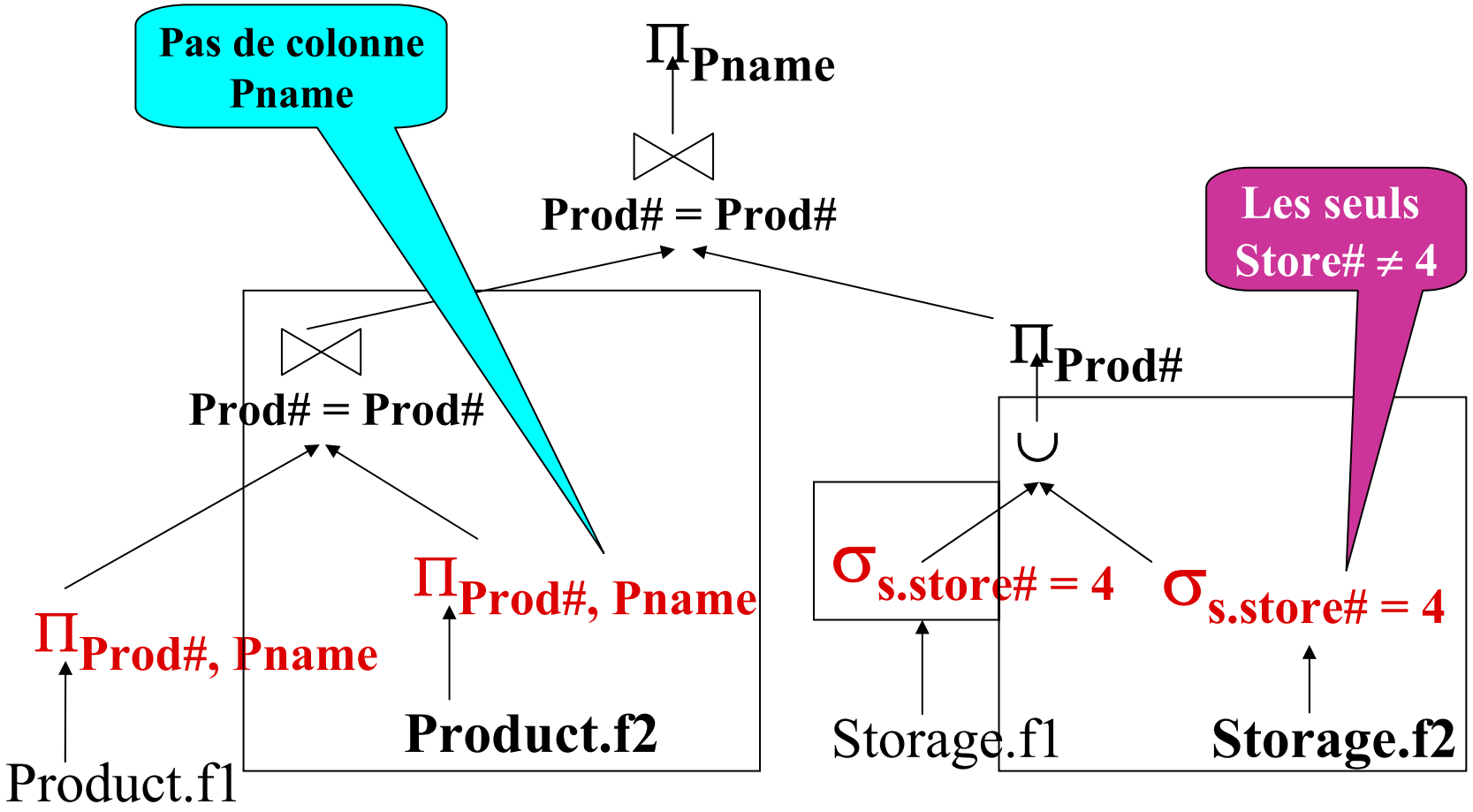




- Seuls les fragments existent ! (⇒ sous-arbres)

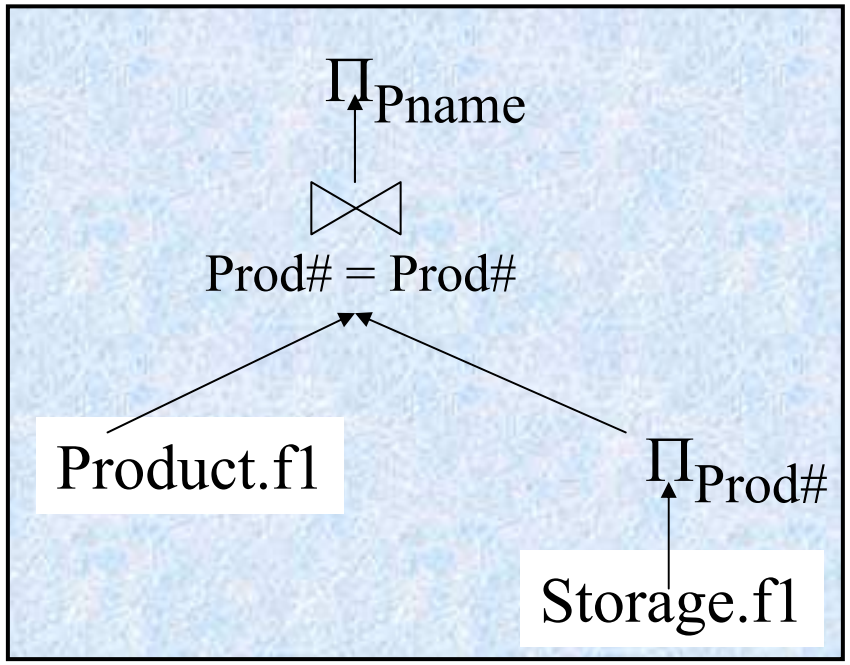


- Descente des sélections et des projections



Bases distribuées : Traitement des Requêtes

- Requête finale :

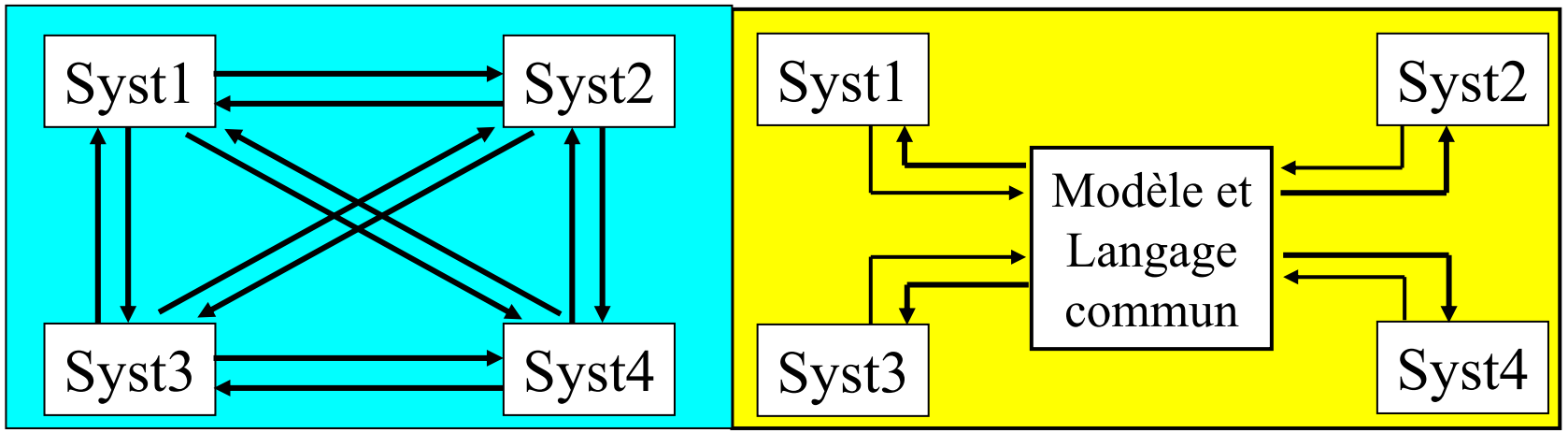


- Accès à 2 sites au lieu de 4 !

Bases distribuées : Traitement des Requêtes

1. *Choisir une stratégie d'exécution* (// joins, semi-join, transfert de données)
2. *Envoyer les sous-requêtes* aux sites impliqués
3. *Chaque site :*
   a. Application optimisation locale
   b. Evaluation
   c. Envoi des résultats au site émetteur
4. *Intégration* des (sous-)résultats

- Cas SGBD hétérogènes
  - *Transparence* → Requête : langage du SGBD local
  - Traduction dans les langages des SGBD cibles
  - Conversion formats des réponses
- Traduction de requêtes; Conversion de données



I.2- Traitement des Requêtes (fin)

- Résultats théoriques (et appliqués)
- Autres travaux d'intérêt :
  - Inclusion de requêtes (généralement DATALOG)
    - $Q1 \subseteq Q2 \Leftrightarrow$ Réponse(Q1) $\subseteq$ Réponse(Q2)
  - Equivalence de requêtes
    - $Q1 \subseteq Q2 \wedge Q2 \subseteq Q1$
  - Déterminer la relation sans évaluation, évidemment

I- Conclusion : Forces et Faiblesses du Relationnel

- Ensemble Simple et Restreint de concepts
- Bases Formelles (Algèbre, Logique)
- Indépendance Données-Programmes
- Langages de Haut Niveau
- Résultats Théoriques
- Systèmes Opérationnels

©Nacer.Boudjlida@loria.fr

Au-delà de la technologie relationnelle

- Limites:
  - « Faible » modèle de représentation de données
    - Contrainte de 1ère forme normale : Relations « plates »
    - Difficile de modéliser des objets complexes
  - Langages de Manipulation
    - Puissance d'expression (→ Langages Hôtes)
    - Types de données User-defined (cf. SGBD extensibles et Objet-Relationnel)

Au-delà de la technologie relationnelle (suite)

- Nouvelle(s) génération(s)
  - Préserver les acquis
  - Pallier les insuffisances
  - Intégrer
    - Technologies émergentes (réseau, client/serveur, Web)
    - Résultats d'autres domaines (programmation, IA, etc.)
  - Répondre aux besoins de nouvelles applications
    - Activités Non Monotones (e.g. conception)
    - Travail en groupe (Groupware) et Coopération

©Nacer.Boudjlida@loria.fr

Plan de la Présentation : Où sommes-nous?

Partie I : Rapide survol de la technologie relationnelle

Partie II : Objets complexes structurés 

Partie III: Web et bases de données 

II: Objets complexes structurés

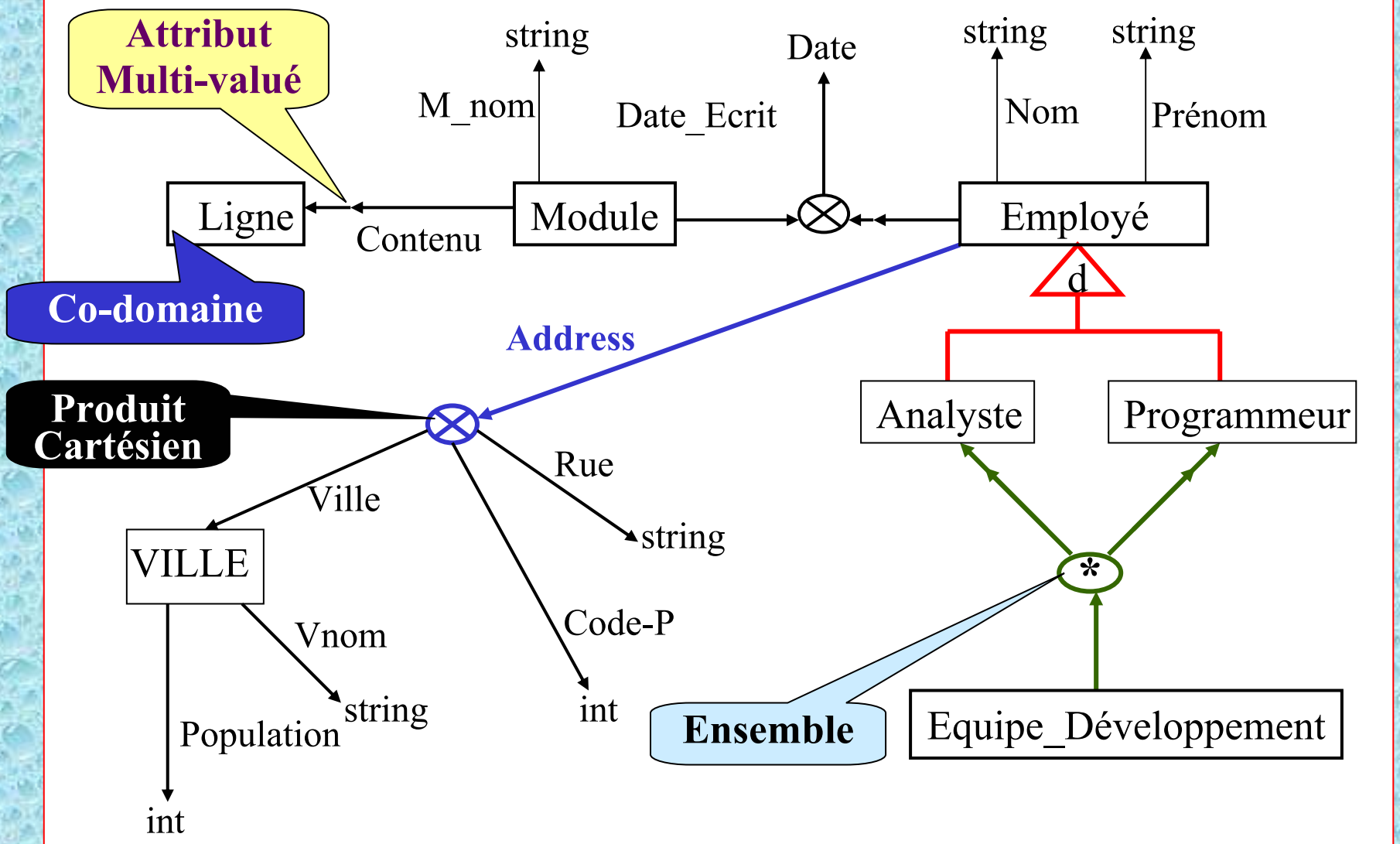
- Approches non « cloisonnées »
  - Logique (DATALOG^{Set})
  - Objets Complexes Structurés :
    - Relationnel : RM/T, NF^2
    - Non relationnels : Modèles Sémantiques
  - Modèles « totalement » Orientés Objets (OMG, ODMG)

- Contenu de la partie II :
  1. Modèles sémantiques
  2. Paradigme objets et bases de données

II.1- Modèles Sémantiques

- Milieu des années 70
- Entité-Relations [Etendu] (Chen)
- Semantic Data Model (Hammer, Mc Leod)
- Functional Data Model
- DAPLEX, etc.
- Objectifs:
  - Etendre le pouvoir d'expression des modèles
  - Réduire/Eliminer l'écart entre le niveau conceptuel et le niveau physique

Modèle sémantique : Exemple



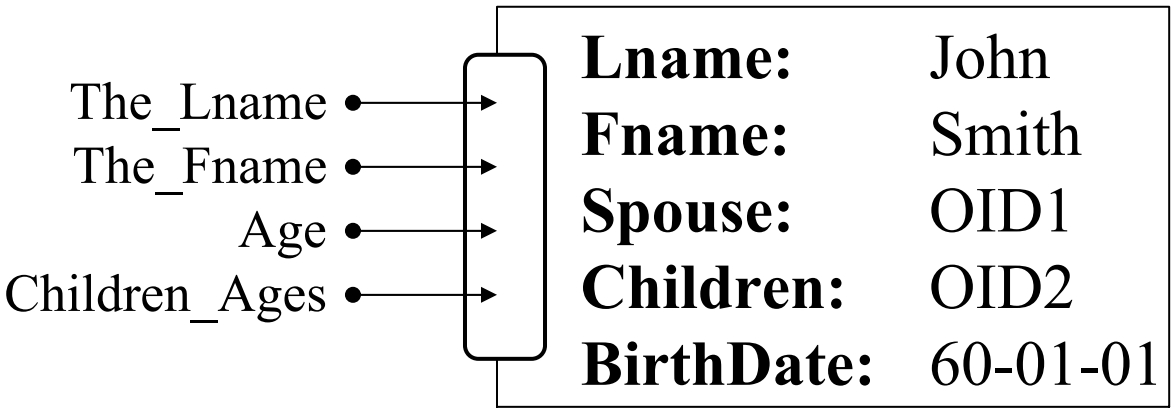
Modèles Sémantiques : Langages et systèmes

- Schéma Conceptuel = Schéma Physique
- Paradigmes:
  - *Impératif*: TAXIS, Dial, Galileo
  - *Functionnel*: Functional Query Language (FDM)
  - *Relationnel*: GEM, DAPALEX, ARIEL
- Systèmes
  - Ré-utiliser l'existant
    - EFDM (Algol persistant); TAXIS (Pascal-R)
  - Mécanismes ad hoc
    - ADAPLEX (ADA + SGF « classique »)

II.1- Modèles Sémantiques : Conclusion

- Pas de fondement formel
- Peu d' implémentations commerciales
- Transition vers les Objets (?)
- Sémantique vs Orienté Objet
  - *Similitude* : Objets complexes imbriqués
  - *Différences* : Modèles et systèmes à objets
    - Comportement des Objets (Opérations/Méthodes)
    - Encapsulation
    - Héritage, polymorphisme, etc.

- Application du paradigme Objet aux BdD
- Objet =
  - Structure (éventuellement complexe):
    - Identité (OID) indépendante de la valeur
    - Etat privé (Valeurs des attributs): atomiques ou OIDs
  - Interface: ensemble de sélecteur de méthodes



Paradigme Objet et Bases de données

- *Classes* : ≈ Extension d'un type d'Objet
- *Héritage* : Attributs et Méthodes
- *Communication par messages*
- *Liaison dynamique et polymorphisme*

Paradigme Objet et Bases de données (fin)

- Orientation Objet "totale" :
  - Gestion d'Objets Complexes
  - Description, Manipulation
  - Sécurité, Intégrité, Confidentialité
  - Accès Concurrents
  - Reprise, etc. $\Rightarrow$ SGBDO
- Objets complexes structurés :
  1. Représentation des objets
  2. Indexation

Représentation des Objets

- Objets et leur description (méta-données)
- Taille des objets
  - Grands ensembles
  - Objets longs
  - Atomes de grande taille (e.g. image, son)
  - Données graphiques, etc.
- Imbrication d'objets
- Index pour objets complexes ?

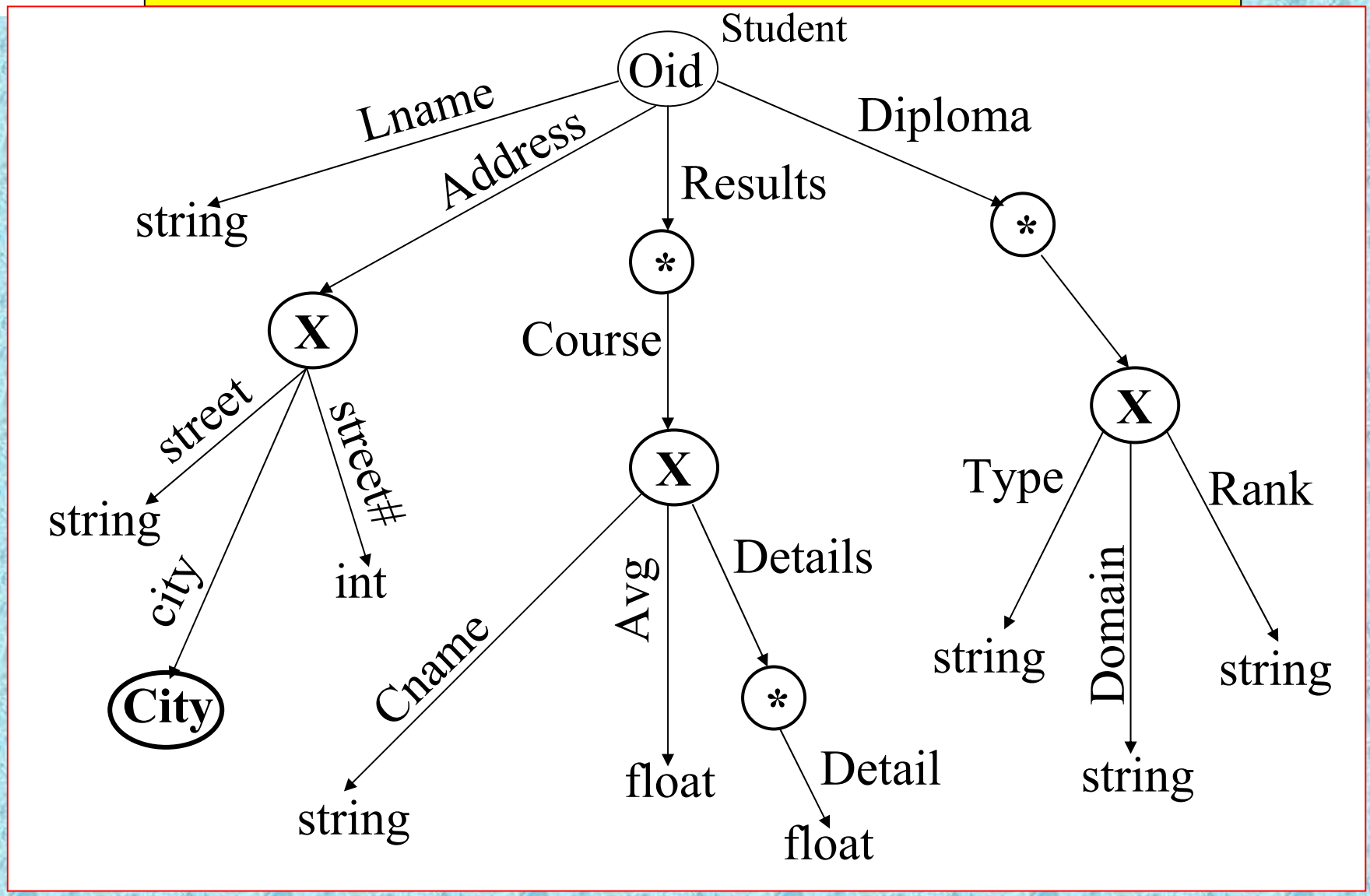
Représentation des Objets

- Multi-Graphe : *une forme parmi d'autres*
  - Feuilles : Valeurs ou Oids
  - Nœuds internes : Constructeurs de types
  - Racine : Object ID
  - Arcs labellés par les noms d'attributs

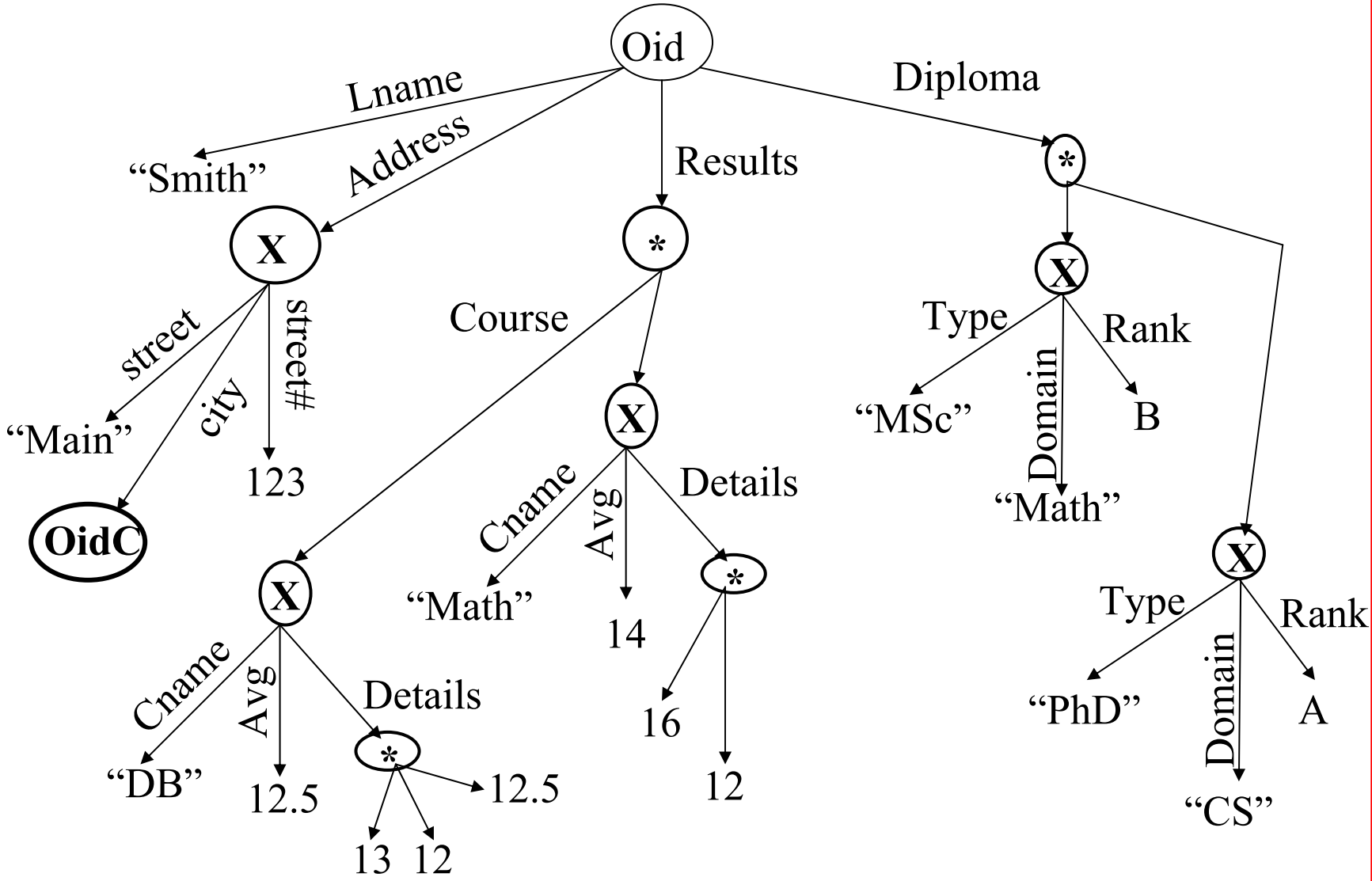
(+) Représentation unique : schéma, instance

(-) Transformation Graphe ↔ « Record »

Représentation des Objets

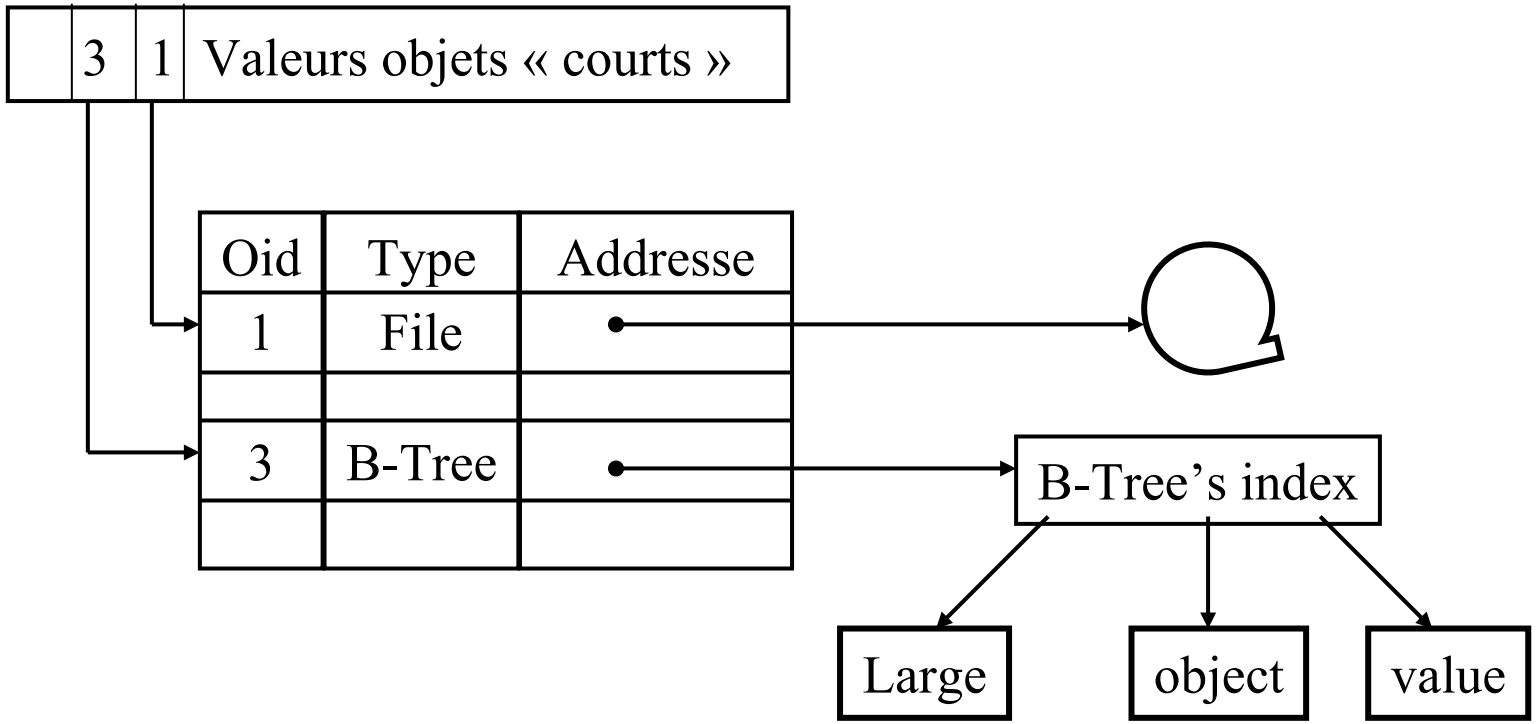


Représentation des Objets



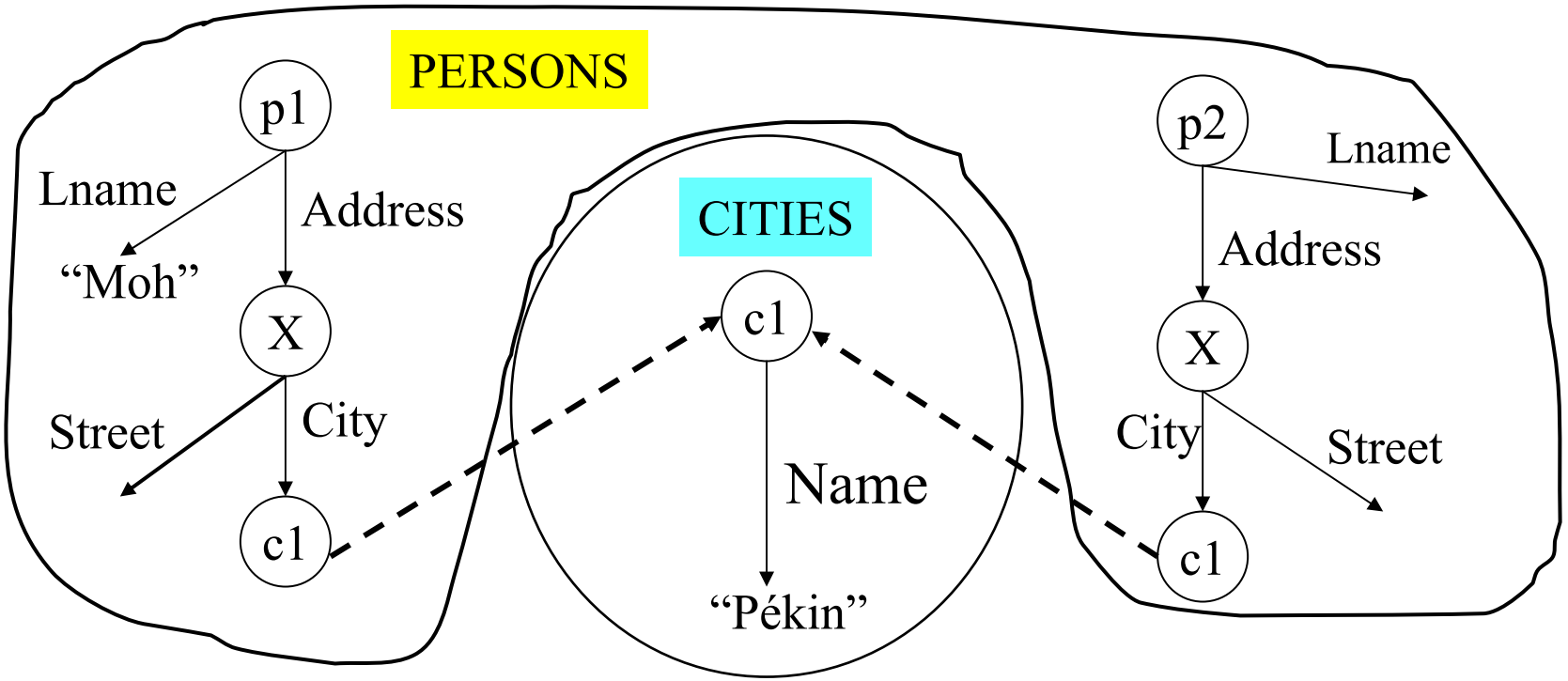
Représentation des Objets Longs

- Objets de petite taille : Valeur immédiate
- Objets longs : « par référence »

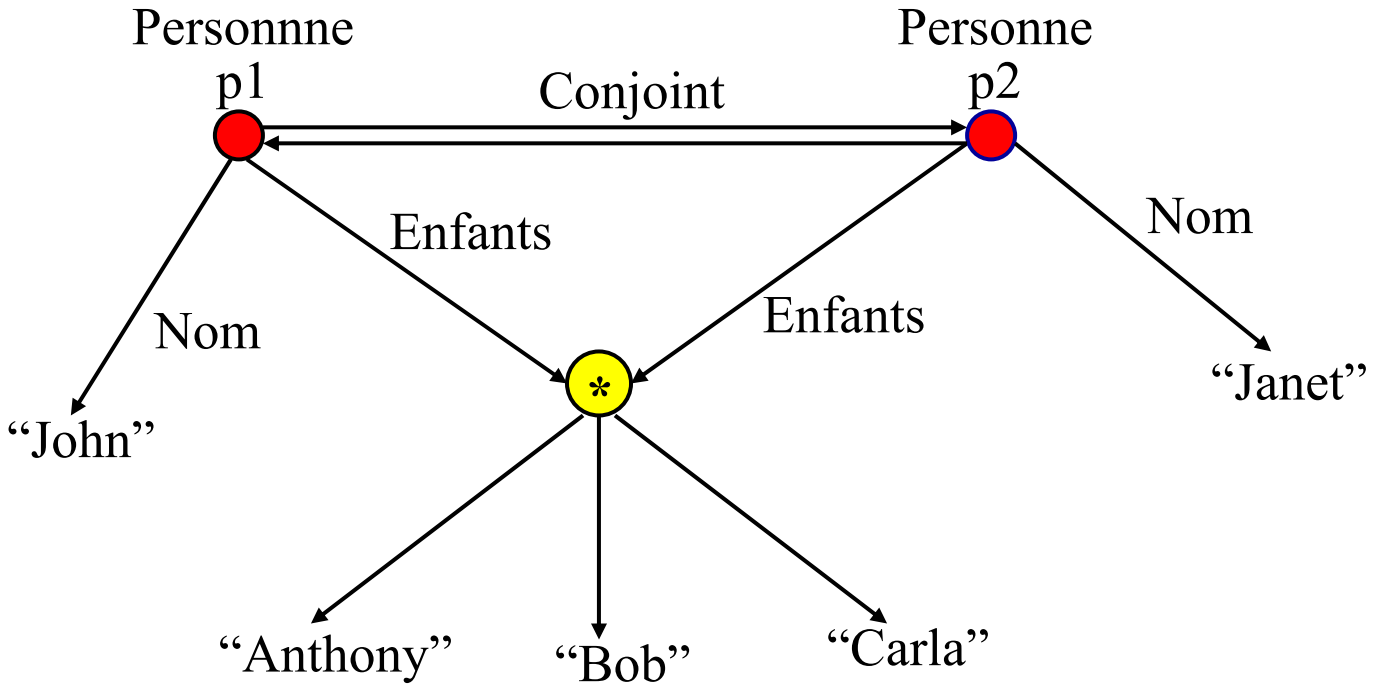


Représentation des Objets et Index

- Index composé : Pékin → {p1, p2}
- Index de chemins (*traversée de classes*)
  - Pékin → {p1, p2}.Address.c1.Name.”Pékin”



- Personnes p1 et p2 ont le même ensemble d'enfants



Paradigme Objet et Bases de données : Egalité

- « Surface » (shallow/value equal)
  - O1: < Nom : 206, Constr : Peugeot, Année : 1999>
  - O2: < Nom : 206, Constr : Peugeot, Année : 1999>
  - Objets *distincts, égaux* en Valeur
- « Profonde » (deep equal)
  - C1: < Nom : 206, Constr : Peugeot, Options: **O1** >
  - C2: < Nom: 206, Constr : Peugeot, Options: **O2** >
  - **O1**: < Type : A1, Valeur : Air conditionné >
  - **O2**: < Type : A1, Valeur : Air conditionné >

Accès aux objets

- Valeur d'un objet :
  - Valeurs des attributs de 1er niveau ?
  - Valeur profonde incluant attributs des sous-classes?
- Lire objet : Représentation base → mémoire
- Ecrire objet : Représentation mémoire → base
- Verrouiller : Objet/Classe ?

II.2- Objets complexes structurés (fin)

- Produits Commercialisés
- Support de persistance (back-ends)
- Tendance :
  - Pas de migration « massive » vers les SGBDO
  - Mais de l'objet « dans » le relationnel

Plan de la Présentation : Où en sommes-nous?

Partie I : Survol du relationnel

Partie II : Objets complexes structurés

Partie III: Web et Bases de données

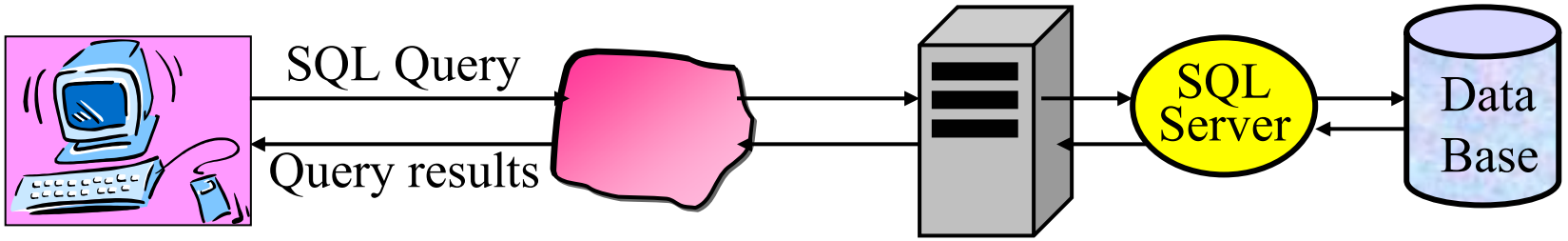
Partie III : Web et bases de données

1. BdD et architectures client/serveur
2. Web et Bases de Données
    1. Le Web en tant que Base de données
    2. Objets semi-structurés
    3. Introduction à XML et sa galaxie

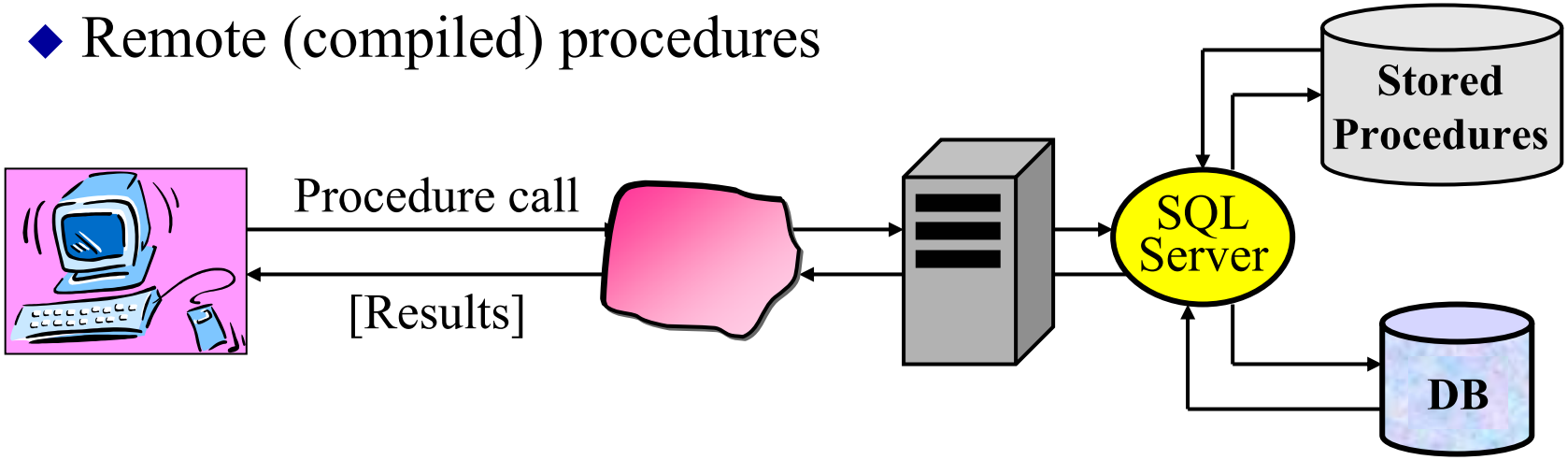
©Nacer.Boudjlida@loria.fr

Serveurs de Données (SQL)

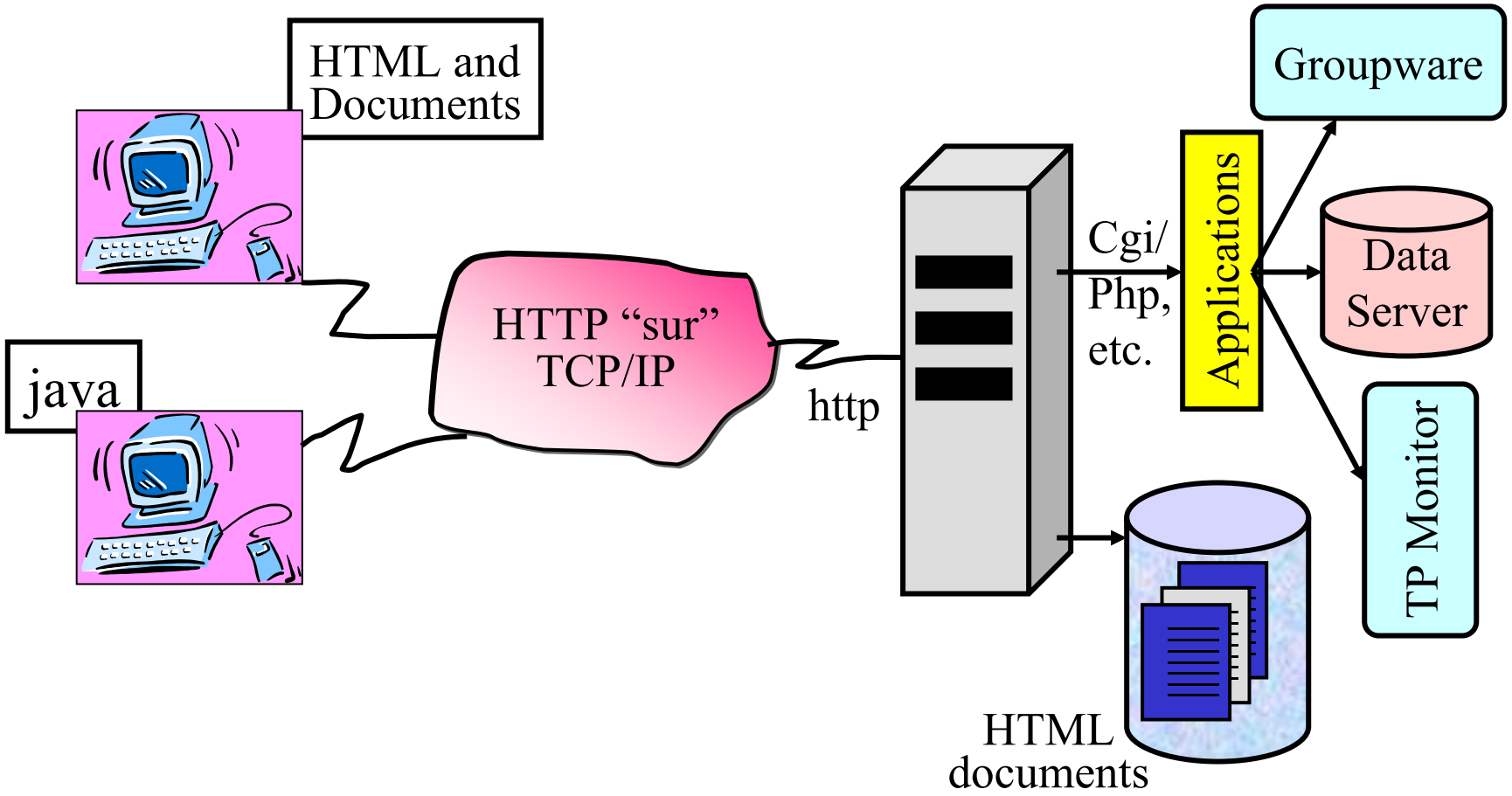
- SQL Queries



- Remote (compiled) procedures

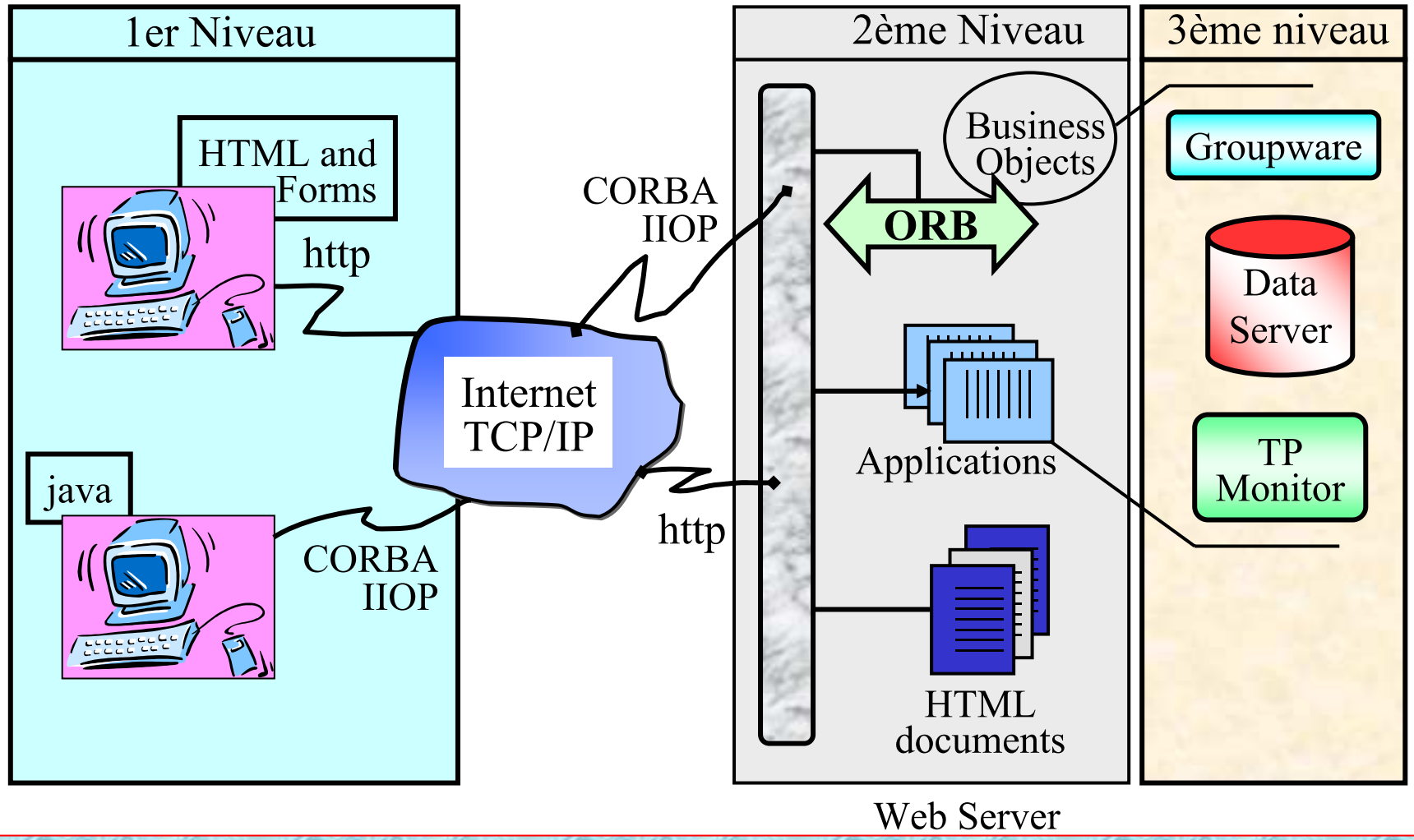


- BdD et Web Serveurs
  - Conversion : Relationnel/Objet ↔ HTML (XML)



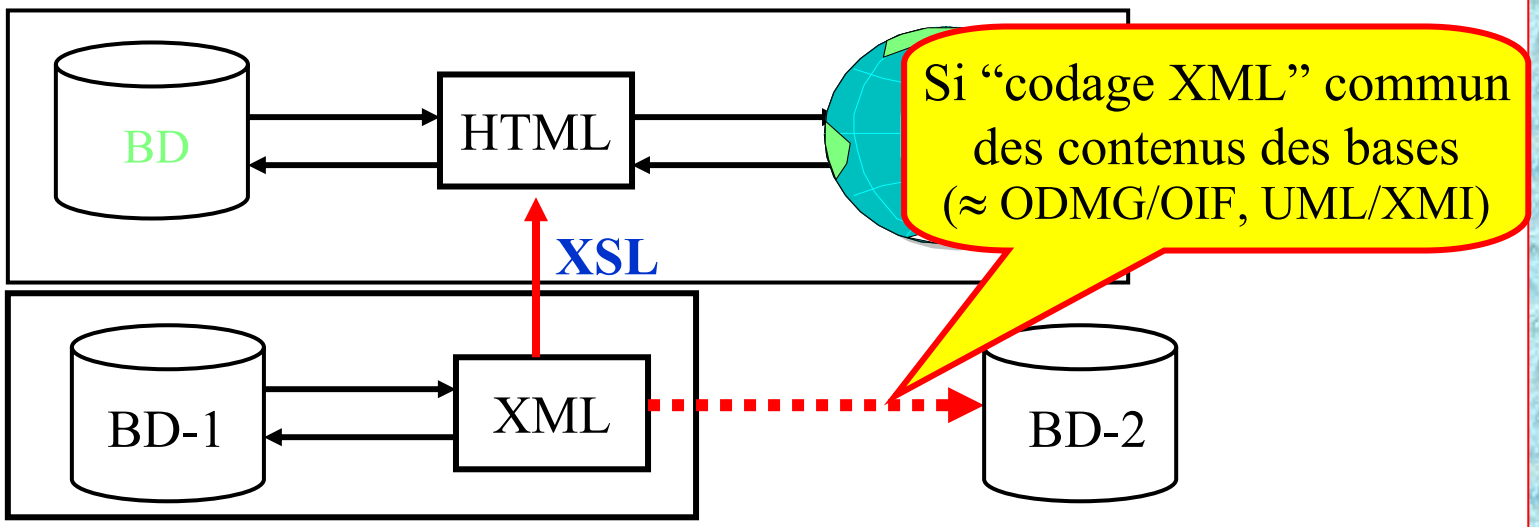
BdD et Architectures Clients/Serveurs (fin)

- Serveur Web-Objet : un modèle d'architecture



Web en tant que BdD

- ≠ Accès à des bases via le Web (cf. types d'architectures Client/Serveur)
  - Langages de scripts (perl, php, etc.)
  - Convertisseurs de formats (médiateurs):
    - Cf. Peer-To-Peer vs Représentation Commune
    - Ce qui est ou n'est pas (commercialement) disponible :

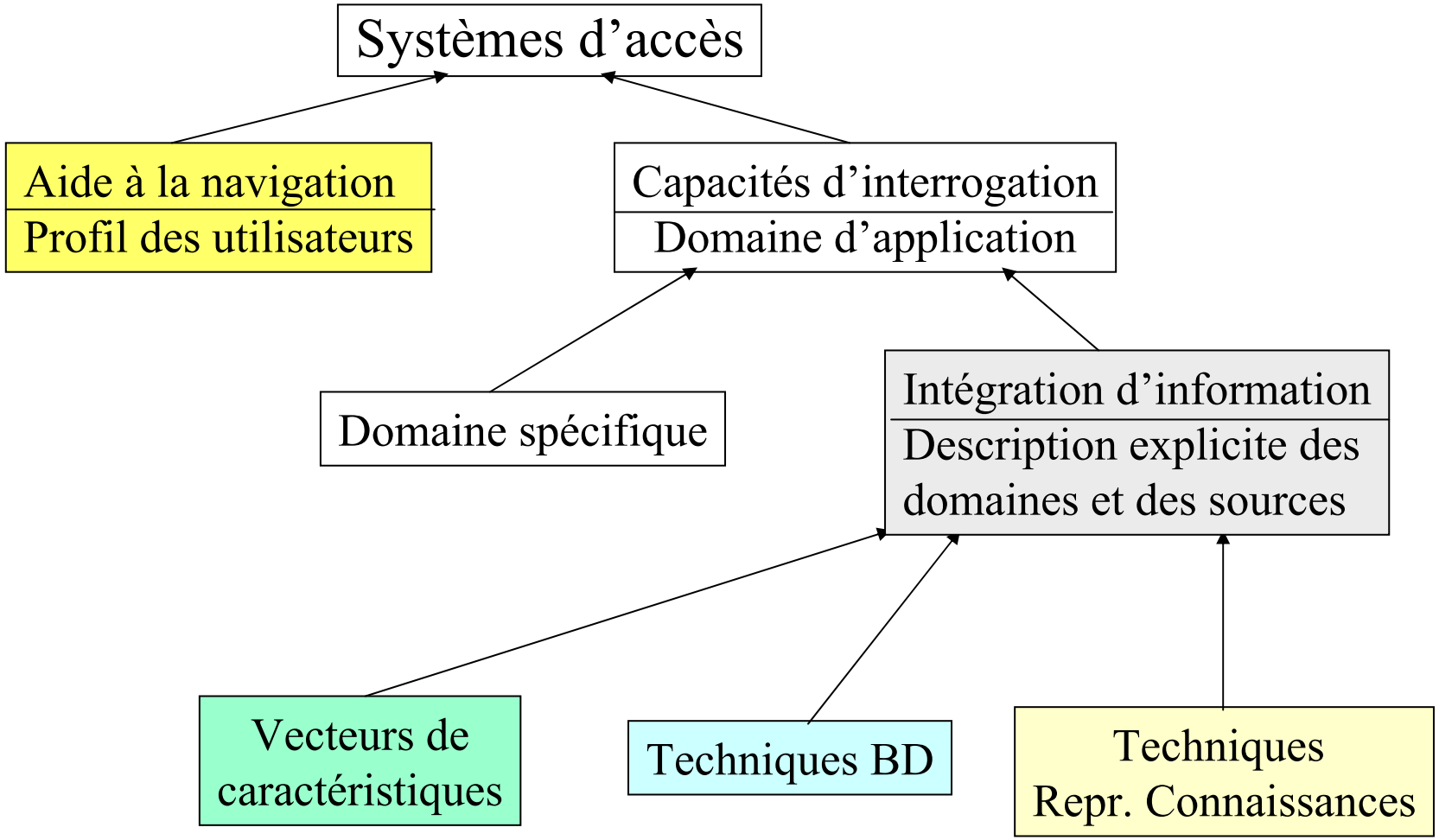


III.2.1. Web en tant que BdD

- Le Web comme base de données : pourquoi ?
  - Sources d'informations
  - Introduction d'un peu « d'organisation »
- Interrogation base de données : schéma nécessaire (!?)
- A priori, aucune structure (Web ≈ gigantesque graphe) : quel(s) modèle(s) de données ?

III.2.1. Web en tant que BdD

- Systèmes d'accès (T. Catarci, ER'99, Paris)



III.2.1. Web en tant que BdD

1. Systèmes d'aide à la navigation (« *surfers* »)
   - Détection profil utilisateur pour l'aider à naviguer

2. Systèmes avec capacités d'interrogation
   - Offrir une vue intégrée de différentes sources
   1. Dédiés à un domaine («*Information Brokers*»)
   2. Généraux : distinction fondée sur le mode de représentation et d'exploitation des informations
      a) Vecteur de caractéristiques
      b) Modèles de données (≈ Bases de données)
      c) Représentation des connaissances

a) Systèmes à vecteur de caractéristiques

- Document caractérisé par un vecteur d'attributs
- Cas des moteurs de recherche (*Altavista, Lycos, etc.)*
- Exploration et indexation des documents
- Recherche par mots-clés

(+) Facilité d'apprentissage

(-) (im)Précision des réponses

b) Systèmes avec modèle de données

- Web ≈ base contenant des informations sur un domaine d'application
  - Intégration des sources: cf. BdD fédérées (*Tsimmis*)
  - Classification et stockage local des informations (*Araneus*)
- Base physique, « virtuelle » (cf. notion de vue) (ou les 2)
- *Médiateurs distribuent les requêtes aux sources*
- *« Wrappers » se chargent des formats des données*

(-) Modélisation des sources

(-) Extraction non automatique de la structure des sources

(-) Evolutivité

c) Systèmes fondés sur la connaissance

- Techniques des SBC pour :
  - Caractérisation de sources
  - Réponses aux requêtes
- Web ≈ Réseau sémantique
- Base de connaissances = Structure de concepts liés par des relations sémantiques (cf. Ontologies)

(-) Temps de réponse

(-) Vocabulaire commun (→ Ontologies)

III.2.1. Web en tant que BdD : Quelques problèmes

- Accès « intelligent » aux informations intimement liés à leur représentation
- Représentations simples (ex: moteurs de recherche) : Efficaces mais inappropriées
- Représentation et structuration des connaissances difficiles mais peuvent contribuer à de meilleurs résultats
- Dynamicité du Web (vie/mort des sites)

III.2.1. Web en tant que BdD : Quelques problèmes (fin)

- Optimisation des requêtes (cf. inclusion de requêtes et « caching »)
- Hétérogénéité et non structuration des sources
- Structures évolutives (cf. dynamicité)
- La suite :
  - Notions de données semi-structurées
  - XML and co. :
    - Langage de description et d'échange
    - Langages d'interrogation pour XML

III.2.2. Données semi-structurées

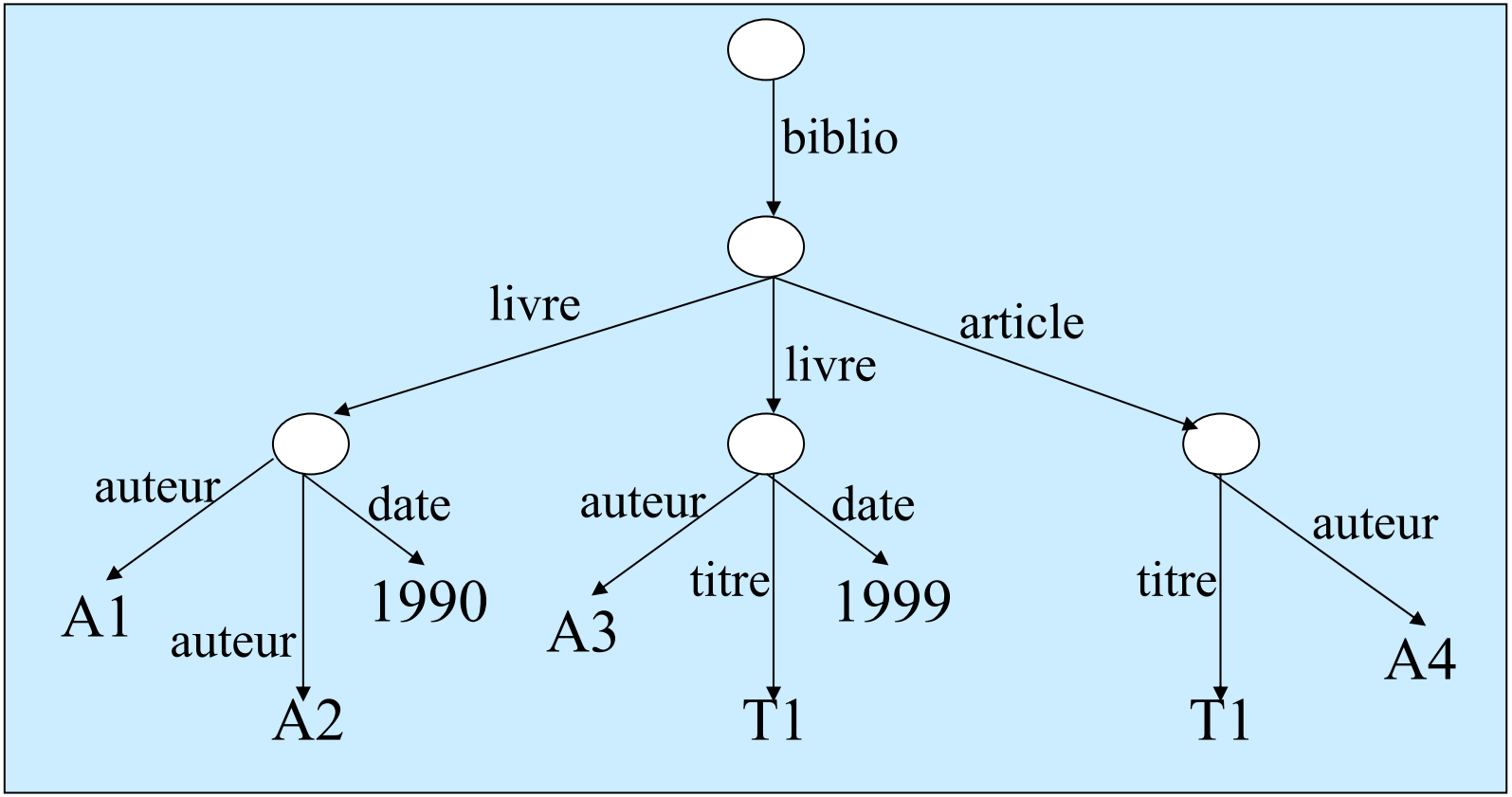
- Notion de données semi-structurées
  - ≈ « Tout ce qui n'est pas relationnel »
  - Pas de langage de description de schéma
  - Données auto-décrites

- Exemples :
  - Bibtex (« rubriques » author, title, year, etc.)
  - Données sur le Web (HTML, XML)
  - Formats d'échange de données (ASN.1, Step/Express, etc.)

III.2.2. Données semi-structurées : Caractéristiques

- Pas nécessairement la même structure
  - Données manquantes (ex: Annotation BibteX)
  - Type varié (ex: date, adresse)
- Structure peut être implicite
- Délibéremment
  - « Oublier » le type de l'instance
  - Sérialiser les valeurs en annotant chaque élément par sa description

- Parcours/Transformation de graphe de données (Récursion structurelle) :

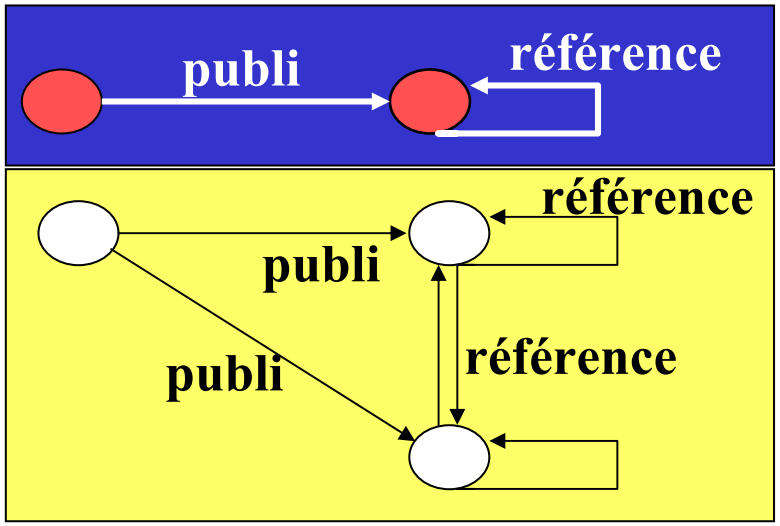


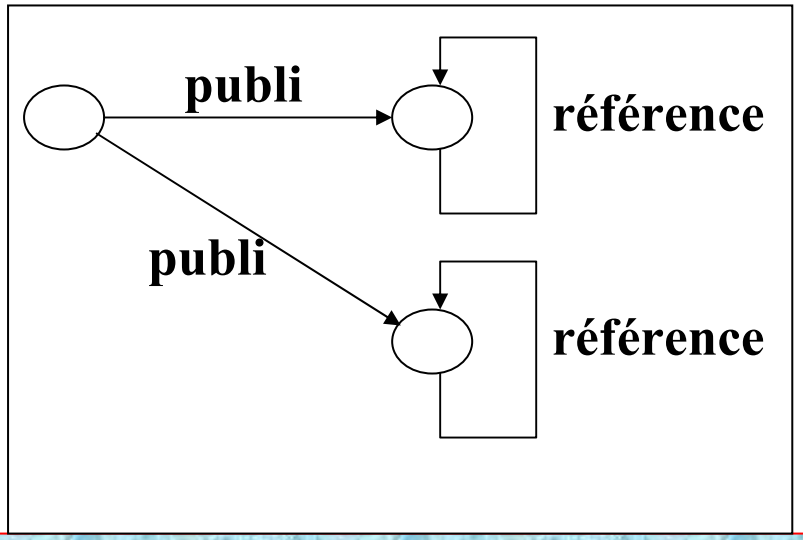
III.2.2. Récursion structurelle

- Transformer tous les entiers en chaînes
  - F(v) = si Entier(v) alors Ent2Chaine(v) sinon v
  - F({}) = {} \\ {} : graphe à un nœud (feuille)
  - F({l : t}) = {l : F(t)}
  - F(t1 ∪ t2) = F(t1) ∪ F(t2) \\ t1, t2 : sous-arbres

III.2.2. Données semi-structurées : Egalité

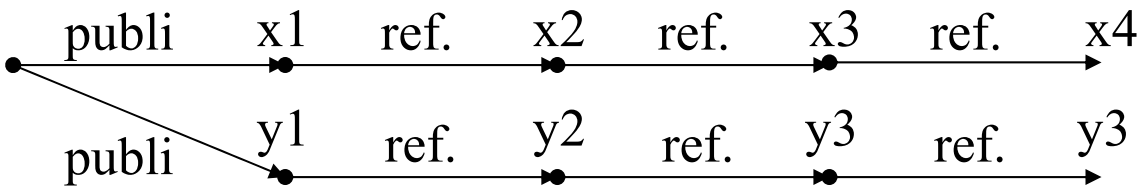
- t1: {livre: {auteur: “A”, auteur: “A”, titre: “T1”}}
- t2: {livre: {auteur: “A”, titre: “T1”}}
- t1 et t2 dénotent le même objet ?
  - Oui si élimination de la redondance
- Cas des données cycliques:





III.2.2. Données semi-structurées : Egalité

- « Déplier » et comparer les arbres



Si x1 = y1, …, xn = yn alors les objets sont égaux

- Comment comparer sans déplier :
  - Calcul d'une *relation de bi-simulation*

©Nacer.Boudjlida@loria.fr

III.2.2. Données semi-structurées : Conclusion

- Structures
  - Irrégulières
  - Implicites (extraction ?)
  - Incomplètes
- Absence ou faible typage
- Exemples : OEM, LOREL
- XML : même veine ?

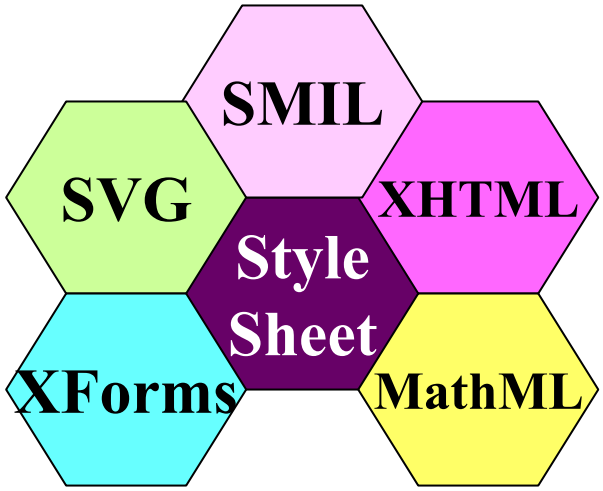
III.2.3. Introduction à XML et sa galaxie

- XML: eXtendible Mark-up Language
  - World Wide Web Consortium (www.w3c.org)
  - Sous-ensemble de SGML (Standard Generalized Mark-up Language) : Gestion Electronique de Documents
  - Description du **contenu** des documents (% HTML)
  - *Objectif Principal* : **Echange** de données
  - Marquage (balisage) « libre » et extensible
  - Hypothèse : *Compréhension commune* des balises (e.g. XMI pour l'échange de diagrammes UML)

III.2.3. Introduction à XML

- Données *a priori* non interprétées (**P**arsed **C**haracter **DATA**) ⇒ Langage facilement analysable
- *Mise en forme du contenu* (XML → HTML)
  - **CSS** (Cascading Style Sheet) : cf. HTML
  - **XSL** (eXtensible StyleSheet Language) : non reconnu comme standard
  - **XSL**Transformation ≈ XSL Microsoft (Internet Explorer)
- XHTML1.0: Reformulation de HTML4 en XML
  - Modules, Document profiles, DTDs (→ XML Schémas), …

- XHTML :
  - XHTML1.0: recommandation, Jan. 2000
  - XHTML2.0: 6ème Draft, Mai 2003
  - *But* : être utilisable avec des balises provenant d'autres « vocabulaires » XML, comme
    - SMIL : Synchronized Multi-Média Integration Language
    - SVG : Scalable Vector Graphics
    - XForms : Formulaires Web
    - OFX : Open Financial eXchange

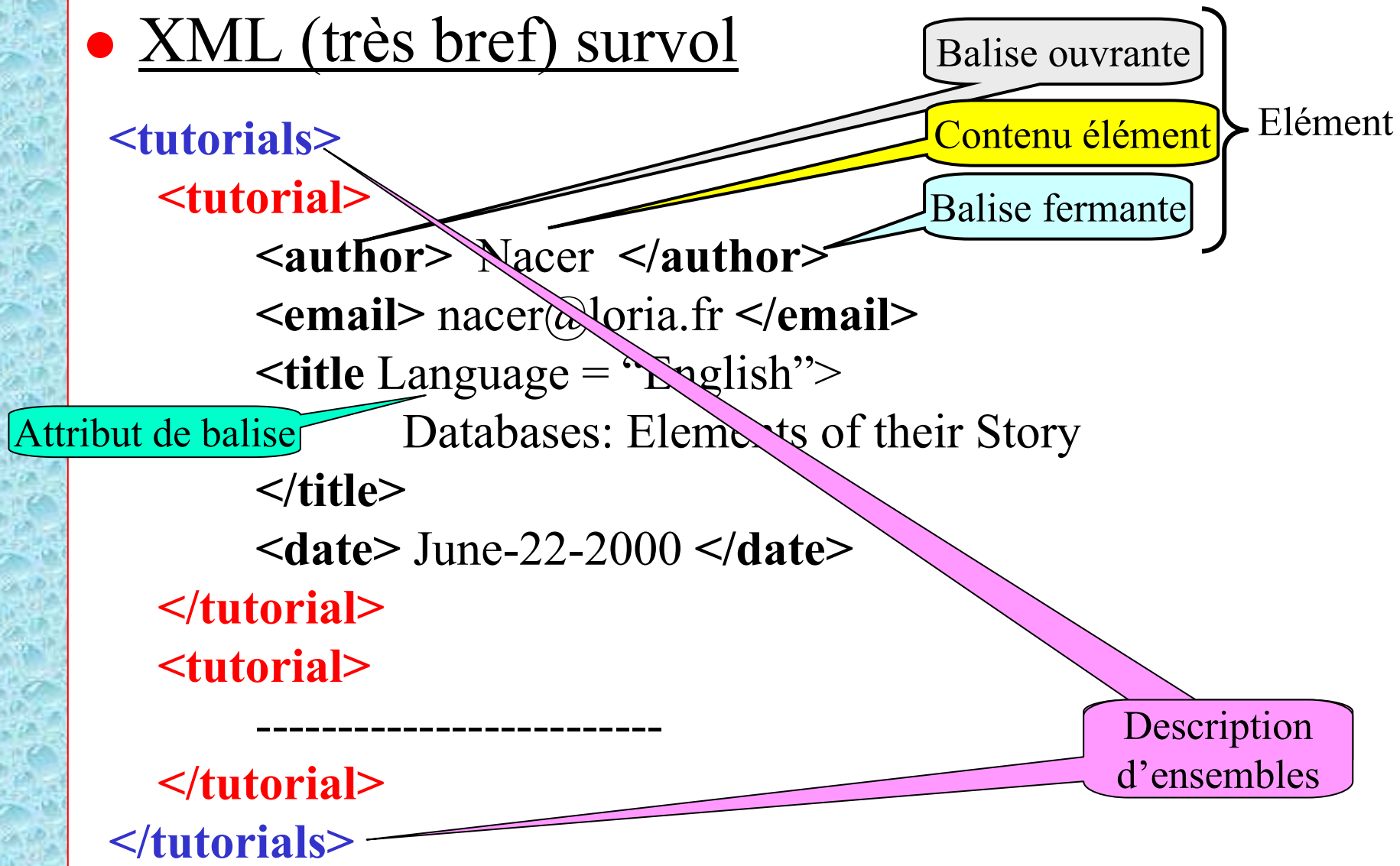


III.2.3.1- Introduction à XML

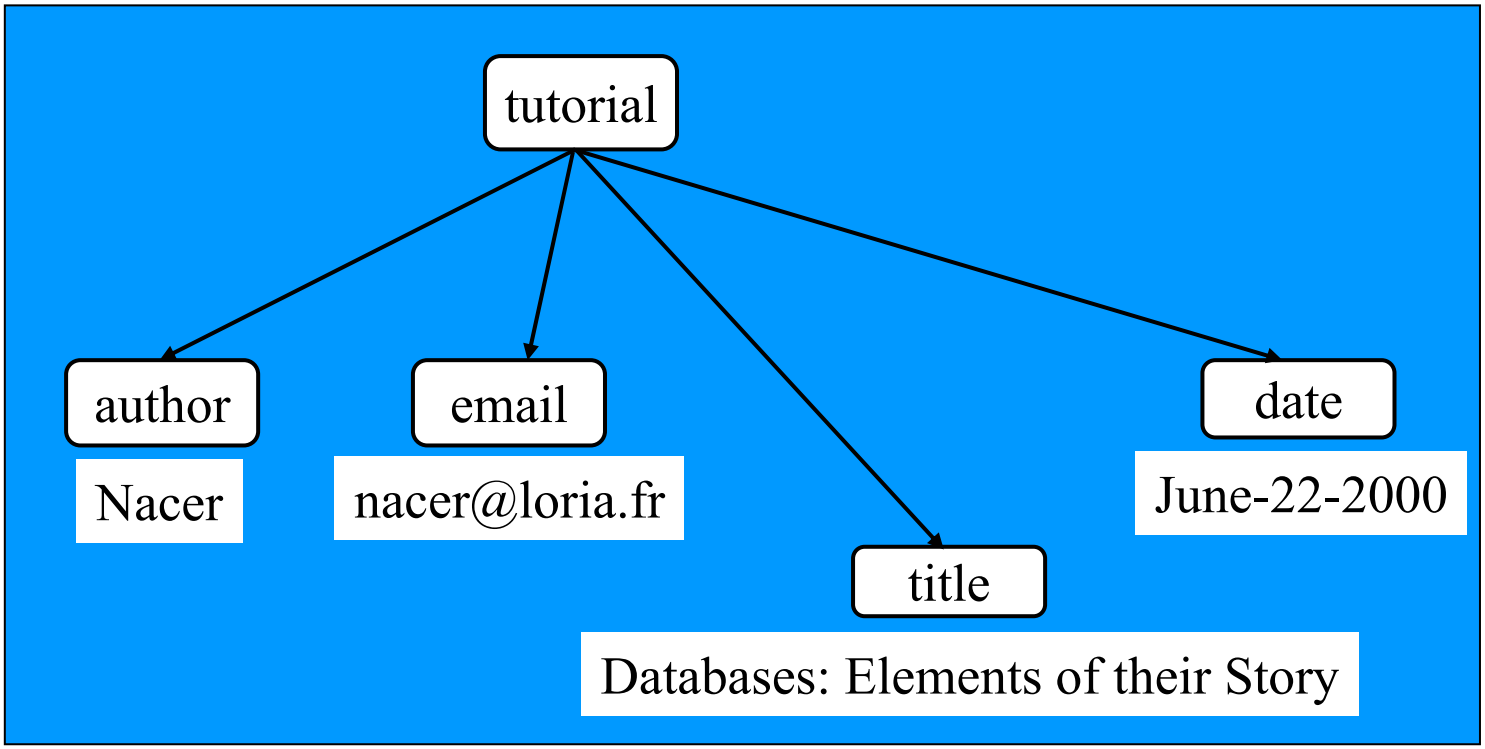
- XML :: HTML
  - Description du contenu
  - Pas d'ensemble prédéfini de balises
  - Support de structures imbriquées ≈ objets complexes (Xlink et Xpointer) :
    - Liens intra et inter-documents
    - Navigation dans des documents XML
  - Données éventuellement auto-décrites :
    - **D**ocument **T**ype **D**efinitions
    - **D**ocument **C**ontent **D**escription (≈ Typage de document)
    - **R**esource **D**escription **F**ramework (pour Méta-données)
    - XML-Schema

III.2.3.1- Introduction à XML

- XML (très bref) survol



- Représentation :



- Document Object Model (DOM) : API
  - Document vu et manipulé comme un *arbre*
  - Indépendance % langages, plateformes
  - Parcourir/agir sur un document
- Simple API for XML (SAX) : Déclenchement d'événement pendant l'analyse
  - DocumentHandler Interface
  - ErrorHandler Interface
  - DTDHandler Interface
  - EntityResolver Interface

- Exemple de structure de document (DTD)
  - Relations person (fn, ln, id), car(plate#, ownerId)

```
<?xml  version "1.0" ?>
<!DOCTYPE  rDB [
<!ELEMENT rDB (person*, car*)>
<! ELEMENT person (fn, ln, id) >
<! ELEMENT car (platenumber, ownerId) >
<! ELEMENT fn (#PCDATA) >
<! ELEMENT ln (#PCDATA) >
<! ELEMENT id (#PCDATA) >
<! ELEMENT platenumber (#PCDATA) >
<! ELEMENT ownerId (#PCDATA) >          ]>
```

- Documents « composites » :

```
<?xml version “1.0” ?>
<!DOCTYPE tutorial [
        <!ENTITY %Scope SYSTEM “D:\TUT1\Tscope.xml” >
        <!ENTITY %Corps SYSTEM “D:\TUT2\Tcorps.xml” >
] >
<tutorial> <meta author = “ …… “, date = “ ……. “ />
        <title> Le Titre </title>
        %Scope;        <!--- Non inclus dans ce document >
        %Corps;        <!--- Non inclus dans ce document >
</tutorial>
```

- **XML Schema** : Définition d'une classe de documents

```
<?xml version ="1.0" encoding="ISO-8859-1"?>
<schema xmlns = "http//www.w3.org/2000/10/XMLSchema"
        targetNamespace =
        "http//www.w3.org/2000/10/XMLSchema">
<element name = "doc" type = "Document"/>
<complexType name = "Document">
        <attribute name = "Requete" type = "varchar"/>
        <element name = "MaTable-1" type = "Table1"/>
        <element name = "MaTable-2" type = "Table2"/>
</complexType>
</schema>
```

- XML :: Données semi-structurées:
  - ≈ Graphe de données auto-décrites
  - ≠ Labels dans les nœuds (:: sur les arcs)
  - ≠ Ordre sur les éléments
  - ≠ Processing Instructions, Commentaires, etc.

- XML : pour l'échange (Constante évolution)

- *Quel(s) langages d'interrogation pour le Web* ?
  - *Génération 1* : type SQL (ex: WebSQL, W3QL)
  - *Génération 2* : Orientation objet (ex: WebOQL)
  - *Génération 3* : Langages pour XML ?

III.2.3.2- Langages d'interrogation pour le Web (1/3)

- *WebSQL*: Web modélisé comme une base relationnelle avec 2 relations (virtuelles) i.e. *graphe d'objets atomiques*
  - Document (url, title, …)
  - Anchor (Base, label, href)
- *Exemple* : « Documents sur Nancy dans les serveurs en Algérie »

```
SELECT d.url, d.title
FROM Document d SUCH THAT d MENTIONS "Nancy"
WHERE d.url CONTAINS ".dz"
```

III.2.3.2- Langages d'interrogation pour le Web(2/3)

- *WebOQL*: Web modélisé comme un graphe d'objets structurés
  - Interrogation :
    - Le Web
    - Une Page
    - Un ensemble de pages liées entre elles
  - Restructuration :
    - HTML-HTML
    - HTML-Base de données
    - Base de données-HTML

III.2.3.2- Langages d'interrogation pour le Web (3/3)

- *XQuery* : Influences
  - *Quilt* : langage de requêtes pour XML
  - *SQL* : Modèle « Select from where »
  - *Xpath 1.0* (*) et *XQL* : Expressions de chemins
  - *XML-QL* : « binding » de variables
  - *ODMG/OQL* : aspect fonctionnel du langage avec expressions imbriquées

(*) Notation pour naviguer dans un document XML

III.2.3.2- Langages d'interrogation pour XML (3/3)

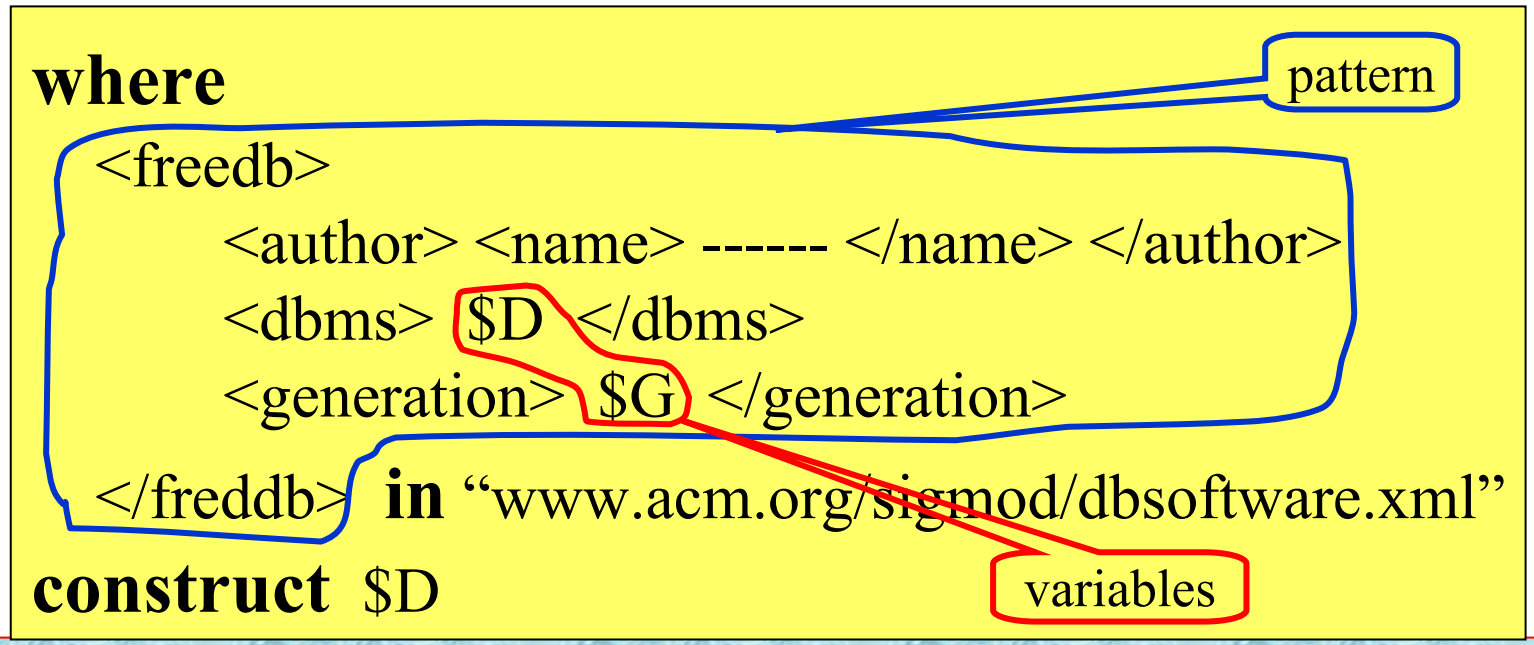
- XQuery : les exigences (« Drafts » → *Mai 2003*)
  - Non procédural
  - Syntaxe en XML et Syntaxe « human readable »
  - Clôture du langage
  - E/S de XQuery : instances de XQuery Data Model
  - Combinaison d'opérateurs y compris des requêtes comme opérandes
  - Possibilité de combiner des données de sources différentes
  - Agrégation et tri

III.2.3.2- Langages d'interrogation pour XML(3/3)

- XQuery : les exigences (requirements)
  - Accès aux méta-données (Schéma, DTD)
  - Extensibilité (types, fonctions)
  - Applicabilité à tout type de sources XML i.e.
    - Documents stockés en mode natif ou
    - Documents vus comme des documents XML (grâce à des médiateurs ou des convertisseurs de formats)
- La suite :
  1. XML-QL : une des influences de XQuery
  2. XQuery : des exemples

III.2.3.2- Langages d'interrogation pour XML : XML-QL

- Relationnellement complet, pour des données de type relationnel
- Principaux composants du langage:
  - **where** ≈ clauses from + where de SQL
  - **construct** ≈ select



III.2.3.2- Langages d'interrogation pour XML : XML-QL

- Intérêt des « patterns » :
  - Spécification du « modèle » de résultat
- Traitement des requêtes :
  - « Comparer » le pattern aux données
  - Lier les variables
  - Retourner les variables de *construct*

- Spécification de la structure du résultat :

```
<query>
  where
    <freedb>
        <author> $A          </author>
        <dbms>   $D </dbms>
        <generation> OO </generatio
    </freedb>  in "www.acm.org/…"
   construct
     <result> <author> $A </author>
              <dbms> $D </dbms>
     </result>
</query>
```

```
<result>
   <author>
     <name> --- </name>
   </author>
   <dbms> --- </dbms>
</result>
        ……..
<result>
        ------
</result>
```


©Nacer.Boudjlida@loria.fr

- Prise en compte des différences de structure
  - Pas le même ensemble de balises
- *Exemple*: <release> non présent partout
  - SGBD et leur release, quand elle existe ?

```
where <freedb> $F </freedb> in "www.acm.org/…",
      <dbms> $D </> in $F
construct
      <result> <dbms> $D </dbms>
            where <release> $R </> in $F
            construct <release> $R </release>
      </result>
```

- Requêtes imbriquées: "SGBD par génération"

```
where
    <freedb>  <generation>  $G  </generation>
    </freedb>  in "www.acm.org/…"
construct
     <result>  <generation>  $G  </generation>
       where <freedb>
                  <dbms>  $D  </dbms>
                 <generation>  $G  </generation>
               </freedb>  in  "----"
       construct    <dbms> $D </dbms>
     </result>
```

- Interrogation des attributs : "SGBD sous Unix"

```
where <freedb platform = "Unix" >
                <dbms> </>  element_as  $D
        </freedb>  in "www.acm.org/…"
construct  $D
```

- **etc.**

III.2.3.2- Langages d'interrogation pour XML : XQuery

- « Drafts » : juin 2001, avril 2002, *mai 2003*
- Principaux types d'expression XQuery
  - Expressions de chemin
  - Expressions For Let Where Return (*FLoWeR*)
  - Expressions avec opérateurs et fonctions
  - Expressions conditionnelles
  - Expressions avec quantificateurs
  - Expressions testant ou modifiant les types de données

- *Expressions de chemins*: prédicats filtrant un ensemble de nœuds
  - document("fichier.xml") : nœud racine
  - . équivaut à self::node()
  - .. équivaut à parent::node()
  - nom équivaut à child::nom (/)
  - @nom équivaut à attribut::nom
  - // équivaut à /descendant ou self::node()/

- ***document***(*"zoo.xml")//chapter[2]//figure[caption = "Tree Frogs"]*
  - Nœud racine : zoo.xml
  - 2ème descendant du nœud racine : <chapter[2]>
  - Recherche des éléments <figure> n'importe où dans l'élément <chapter>
  - Sélectionner ceux ayant un attribut <caption> avec la valeur "Tree Frogs.".

- ***document***(*"zoo.xml")//chapter[2* ***TO*** *5]//figure*

- *Chemins de localisation*

  Syntaxe : **indicateur de relation::filtre[prédicat]**

  - Troisième enfant : **child::*[3]**
  - Dernier descendant: **descendant::[position()=last()]**
  - Films sans acteur : **film[count(child::acteur)=0]**
  - Film dont l'attribut titre est égal à Brazil : **film[child::@titre='Brazil']**

III.2.3.2- Langages d'interrogation pour XML : XQuery

```
<DesLivres>
   <book year = "1994">
        <title>TCP/IP Illustrated</title>
        <author>  <last id =12>Stevens</last>
                  <first>W.</first>      </author>
        <publisher>Addison-Wesley</publisher>
   </book>

   <book year = "1998">
        <title>TCP/IP Illustrated< /title>
        <author>  <last id = 12>Stevens</last>
                  <first>W.</first></author>
        <publisher> Morgan Kaufmann </publisher>
   < /book>
</DesLivres>
```

- **[FOR | LET] … WHERE … RETURN**
  - ***FOR** $b **IN** **document**("bib.xml")//book*
    ***WHERE** $b/publisher = "Morgan Kaufmann"*
    ***AND** $b/year = "1998"*
    ***RETURN** $b/title*
  - ***LET** $b := **document**("bib.xml")//book*
    ***RETURN** $b/title*
- [unordered()|distinct()]

- *Quantificateurs* : exemple

```
<items>
    <item_tuple>
        <itemno>1001</itemno>
        <description>Red Bicycle</description>
    </item_tuple>
    <item_tuple>
        <itemno>1002</itemno>
        <description>Red Bicycle</description>
    </item_tuple>
    <item_tuple>
        <itemno>1003</itemno>
        <description>Red Bicycle</description>
    </item_tuple>
</items>
```

- *Quantificateurs*: **SOME | EVERY … IN … SATISFIES**

```
FOR $i IN document("data/R-items.xml")//item_tuple
WHERE NOT(
      SOME $b IN document("data/R-bids.xml")//bid_tuple
      SATISFIES $b/itemno = $i/itemno)
RETURN
      <no_bid_item>
            {$i/description}
      </no_bid_item>
```

- Jointure

***FOR** $i **IN document**("catalog.xml")//item,*

*$p **IN document**("parts.xml")//part[partno = $i/partno],*

*$s **IN document**("suppliers.xml")//supplier[suppno = $i/suppno]*

RETURN

- etc.
- Mais quelle sémantique *formelle* ? (cf. Use cases)

Conclusion : Bases de données vs Web

- Bases de données
  - Distinction entre les niveaux logiques et physiques
  - Données structurées et fortement typées (schémas)
  - Langages de description et de manipulation
  - Concurrence, etc.
- Sources Web:
  - Données non ou semi-structurées (“typage faible”)
  - (Souvent) pas de schéma (méta-données)
  - Pas (encore) de langage d’interrogation