

# Gestion de données réparties

# Architectures Réparties

- Définition et motivation
- Conception
  - Fragmentation
  - Réplication
- Traitement des requêtes

# Motivations

- Limites des architectures centralisées :
  - Données des entreprises disséminées sur plusieurs sites (banques, sociétés, compagnies de transport,...)
  - Goulot d'étranglement
  - Développement des réseaux
  - Besoin de fédérer les bases de données

# Principes

- Coopération entre plusieurs bases :
  - BD locales, avec des accès locaux rapides, et des accès aux autres SGBD du réseau (accès globaux).
- Plusieurs niveaux d'intégration :
  - Client/serveur : BD centralisée, seuls certains traitements (interface, p.ex.) sont locaux.
  - Accès distant (Remote Data Access) : accès simultané à plusieurs bases de données locales
  - Vues réparties : extension du mécanisme de vues pour définir des vues sur plusieurs sites.

# Définitions

- Bases de données interopérables
  - Coopération, mais pas d'intégration.
  - Les bases sont homogènes ou hétérogènes
- Multibases
  - BD hétérogènes pouvant interopérer via un langage commun
- Bases de données fédérées
  - BD hétérogènes intégrées via un modèle commun et un langage commun (couplage fort)

# Définitions

- Bases de données réparties
  - Plusieurs bases sur plusieurs sites, mais une seule BD « logique ».
  - Les ordinateurs (appelés sites) communiquent via le réseau et sont faiblement couplés.
  - Chaque site contient des données de la base, peut exécuter des transactions locales et participer à l'exécution de transactions globales
  - La répartition affecte les données, les traitements, les contrôles
  - La topologie des BD réparties est semblable à celles des réseaux : anneau, étoile, arbre, ...

# Evaluation

- avantages
  - extensibilité
  - partage des données hétérogènes et réparties
  - performances avec le parallélisme
  - disponibilité avec la réplication
- inconvénients
  - administration complexe
  - complexité de mise en oeuvre et de développement
  - distribution du contrôle
  - difficulté de migration
  - surcharge (l'échange de messages augmente le temps de calcul)

# Evaluation

## Paramètres à considérer

- Coût et temps de communication entre deux sites
- Fiabilité
  - fréquence des pannes des sites, du réseau
- Accessibilité aux données
  - accès aux données en cas de panne des sites, du réseau.
- Accès aux sites les moins encombrés, les plus puissants

# SGBD distribué

Objectif : Rendre la distribution (ou répartition) des BD locales "transparente"

- Indépendance des données

On interroge sans savoir où sont localisées les données

- catalogue des données réparties

- Atomicité des transactions

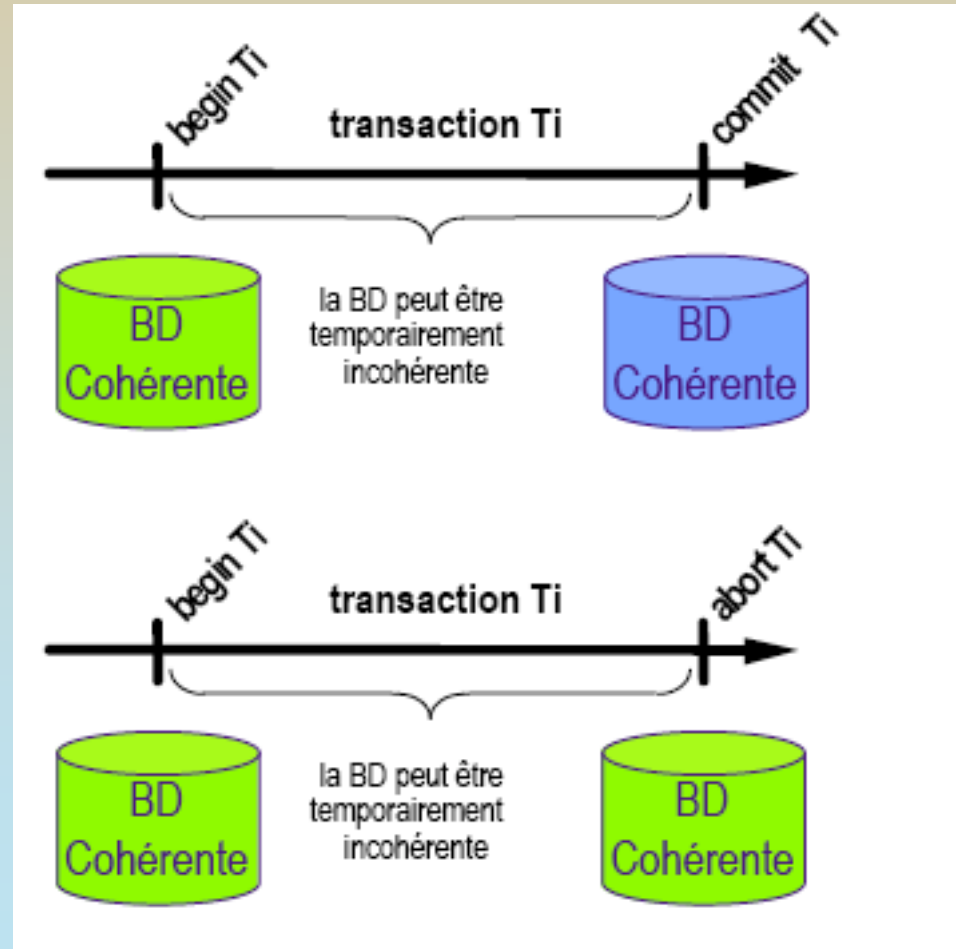
- traitement des requêtes distribuées

- gestion de transactions distribuées

- maintien de la cohérence et de la sécurité

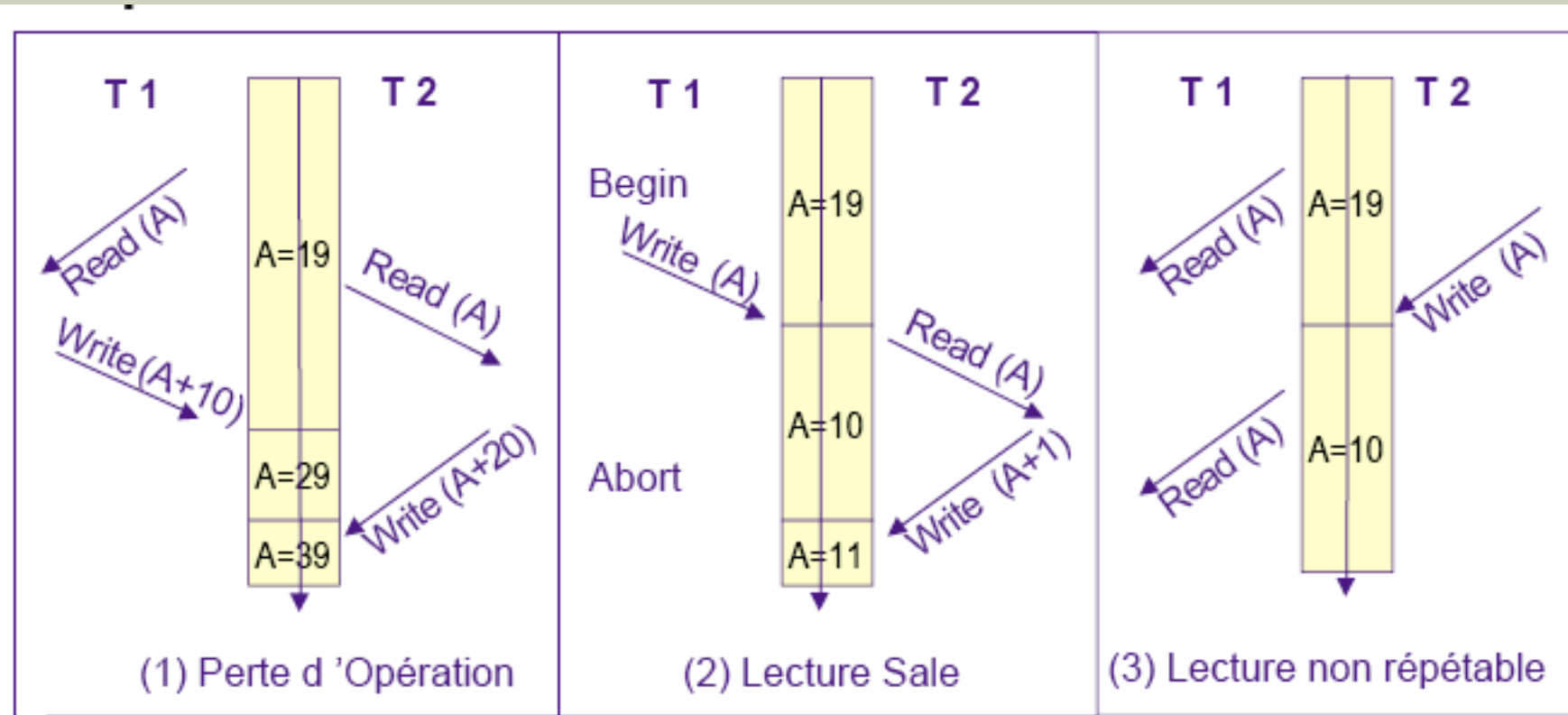
# Rappel transactions

- Une *transaction* = séquence d'instructions qui soit réussit, soit échoue.
- Les transactions doivent posséder les propriétés ACID :
  - A = atomicité (une transaction est complète ou abandonnée complètement )
  - C = consistance (une transaction doit transformer le système d'un état consistant à un autre état consistant)
  - I = isolation (chaque transaction est vue indépendamment des autres )
  - D = durabilité (les transactions validées ont des effets permanents)



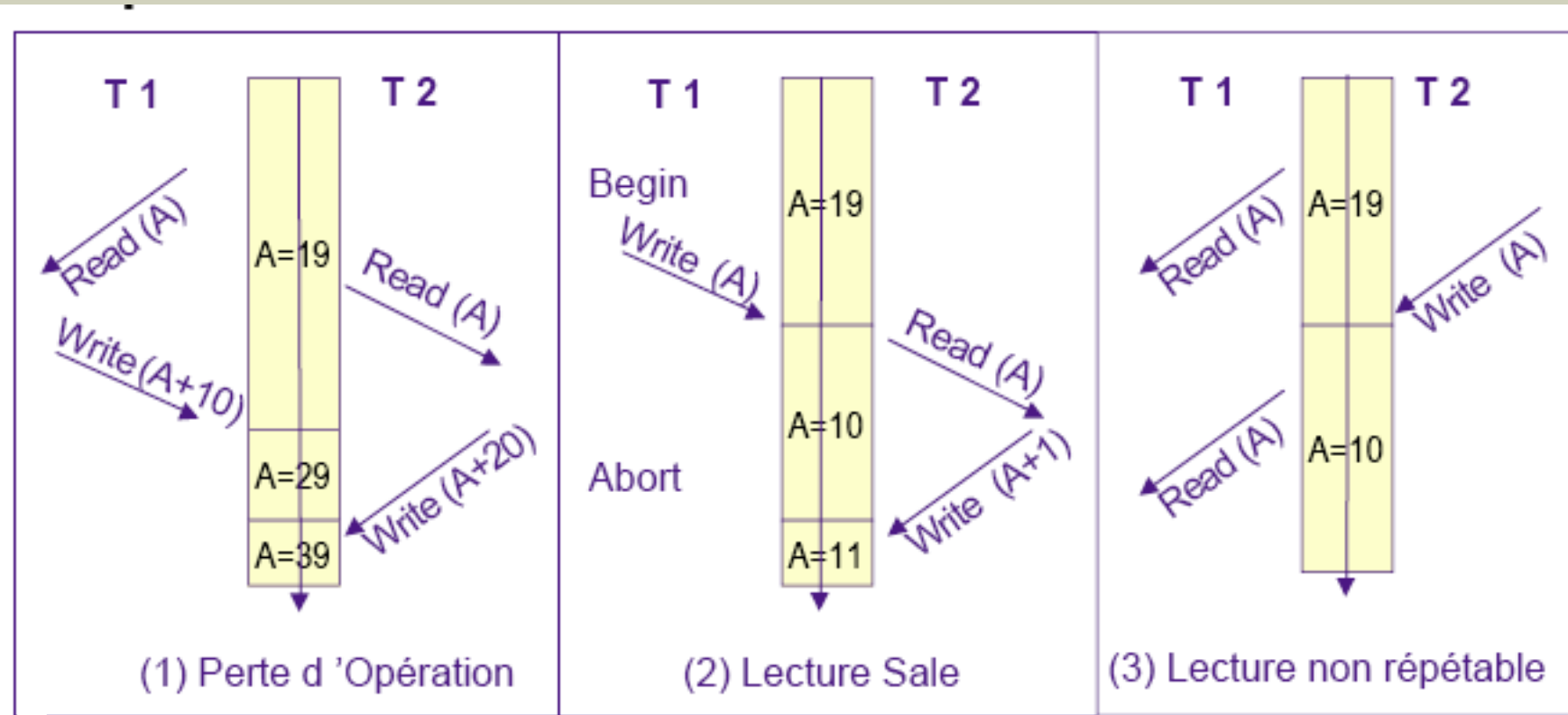
# Rappel transactions

- Une *exécution concurrente* non contrôlée de plusieurs transactions peut amener des incohérences.



# Rappel transactions

- Une *exécution concurrente* non contrôlée de plusieurs transactions peut amener des incohérences.



# Rappel transactions

- Contrôle de concurrence
- Sérialisabilité
  - Entrelacement des actions tel que le résultat des actions est équivalent à une exécution séquentielle.
- Méthodes
  - Pessimistes
  - Optimistes

# Rappel transactions

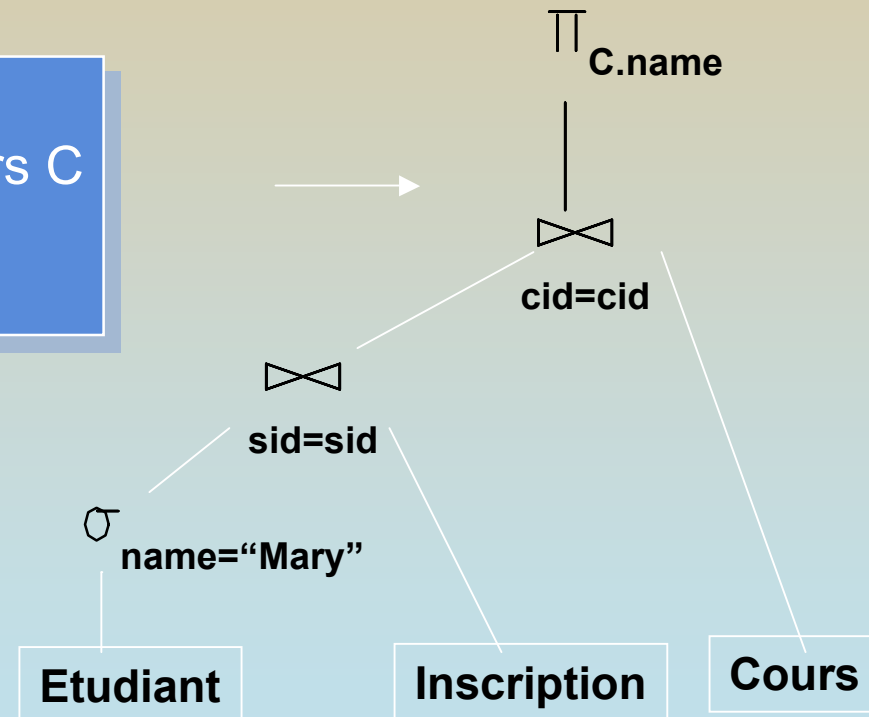
## Niveau d'isolation

	Lect sale	Lect N rép	Fantômes
• READ-Uncommitted	O	O	O
• READ-Committed	N	O	O
• REPETABLE READ	N	N	O
• SERIALIZABLE	N	N	N

# Rappel requêtes

*Requête Déclarative SQL*  $\longrightarrow$  *Plan d'exécution*

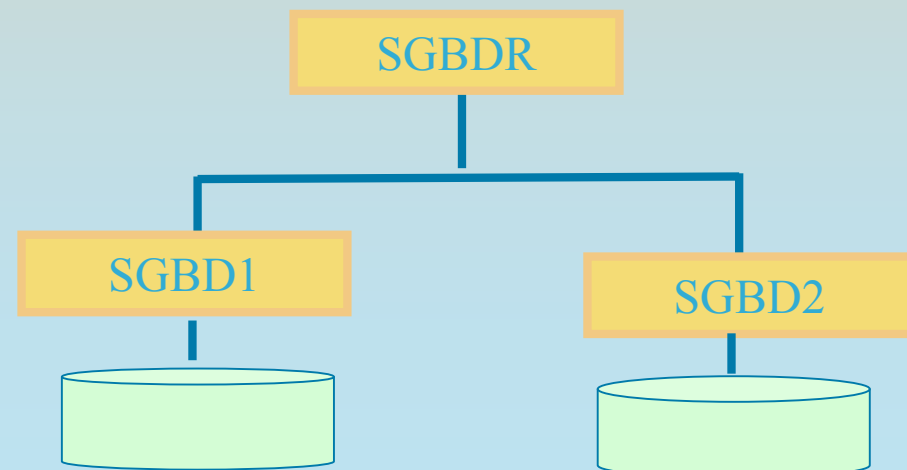
```
SELECT C.name
FROM Etudiants E, Inscription I, Cours C
WHERE E.name="Mary" and
      E.ssn = I.ssn and I.cid = C.cid
```



L'optimiseur choisit le meilleur plan d'exécution pour la requête

# SGBD distribué

- Même SGBD, plusieurs serveurs
  - Système distribué Homogène
- Divers SGBD Multibases système (hétérogène)



# Evaluation

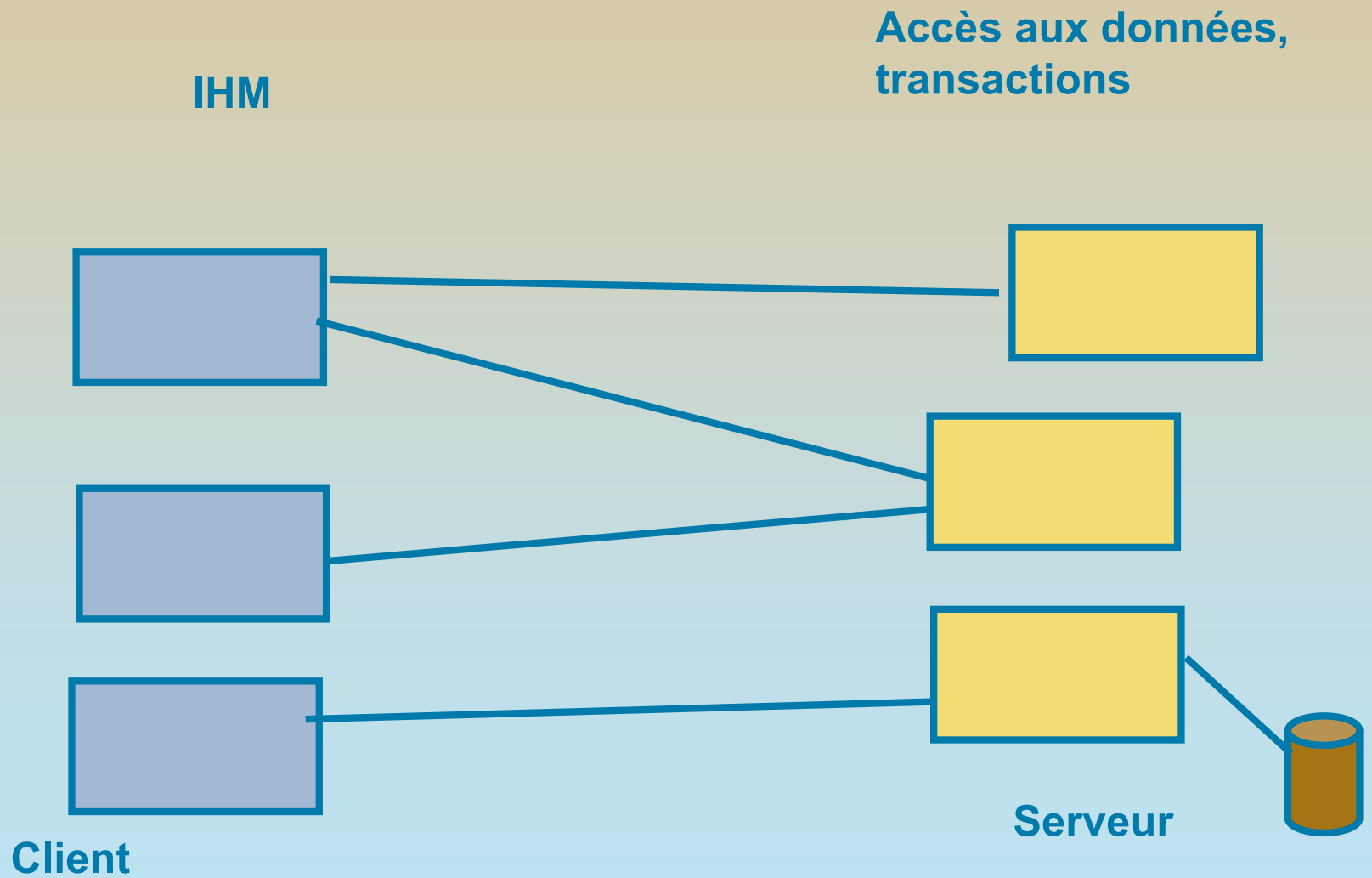
- Avantages
  - extensibilité
  - partage des données hétérogènes et réparties
  - performances avec le parallélisme
  - disponibilité (tolérance aux pannes) avec la réplication
- Difficultés
  - administration complexe
  - distribution du contrôle
  - difficulté de migration

# SGBD distribué

## Architectures distribuées

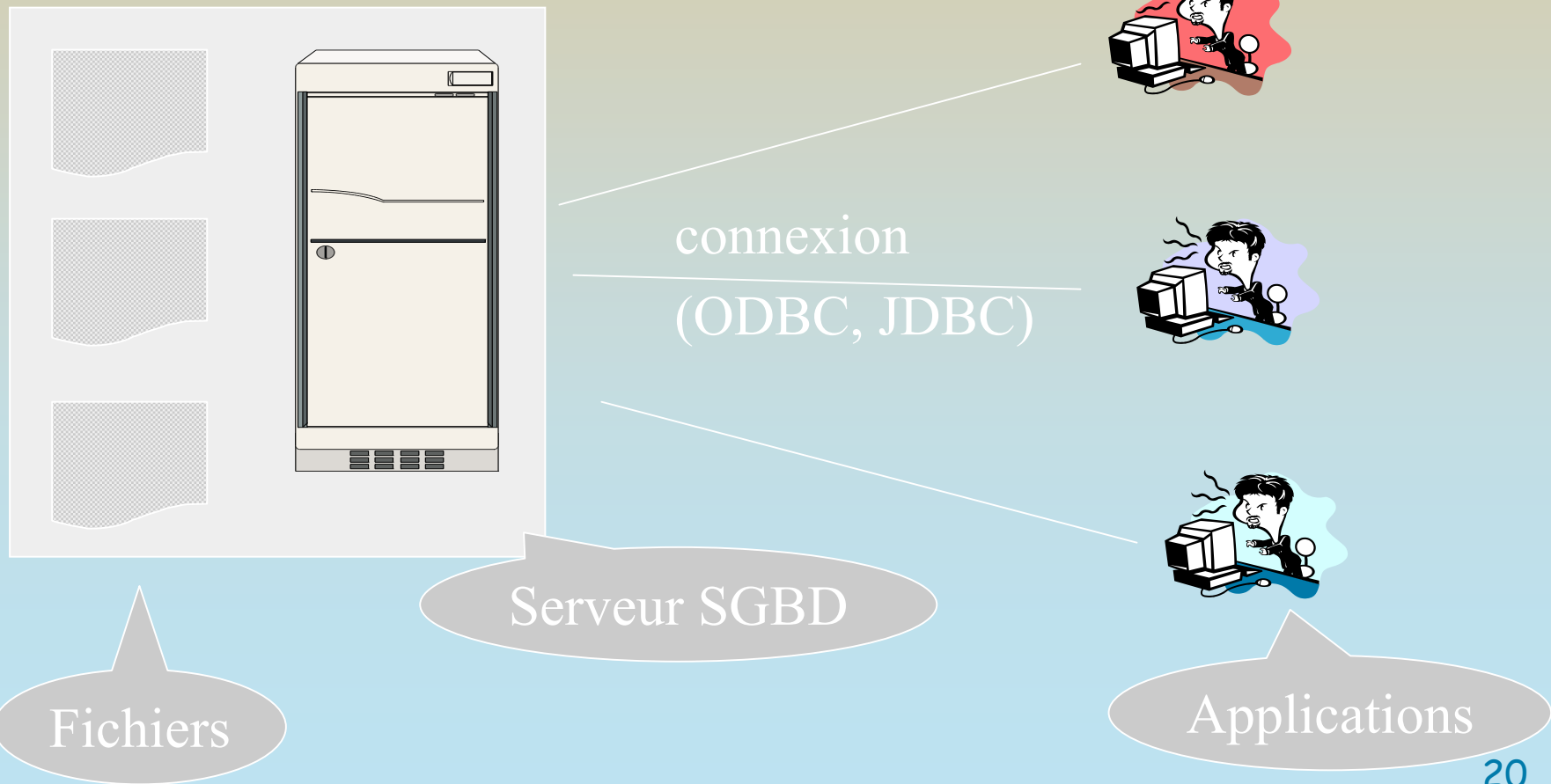
- Client-Server
- Collaborating Server
- Middleware

# Client-Serveur

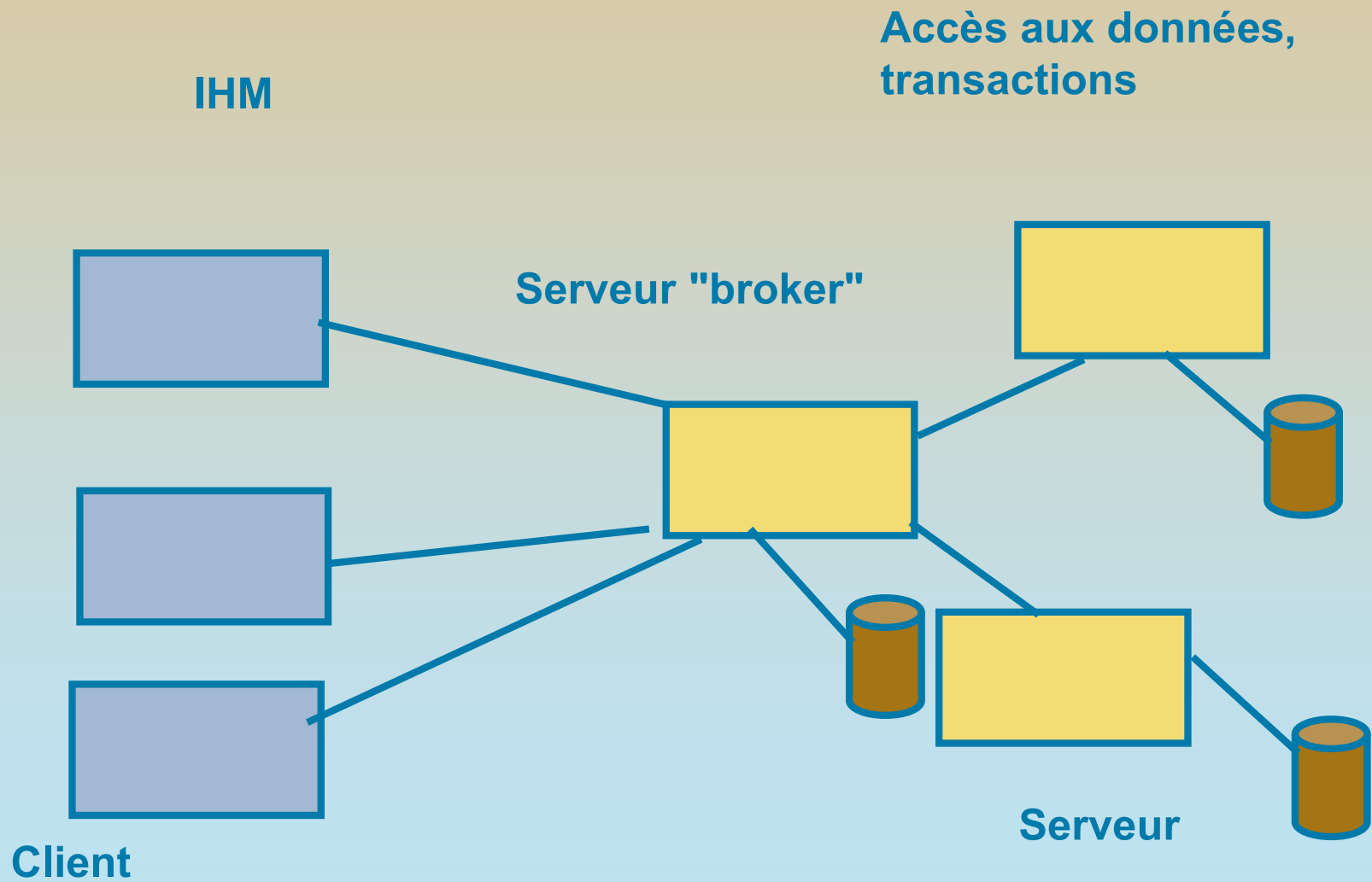


# Client-Serveur

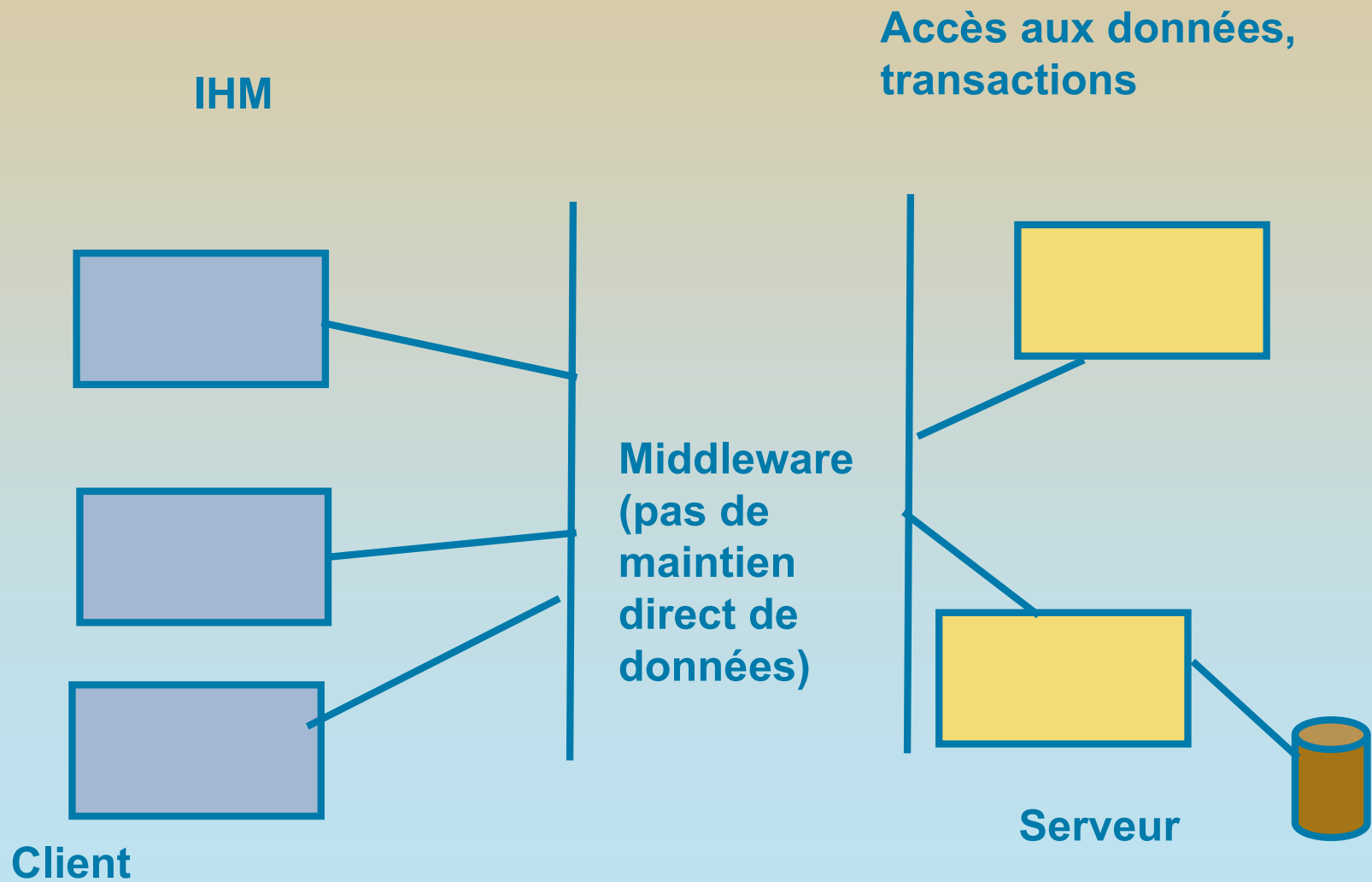
“2 tier system” ou “client-server”



# Collaboration Server Systems

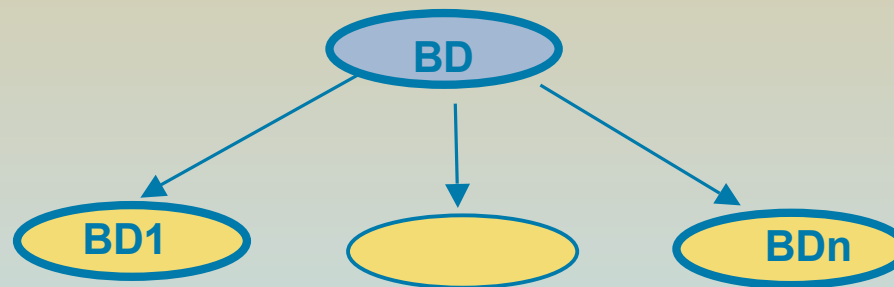


# Middleware Systems

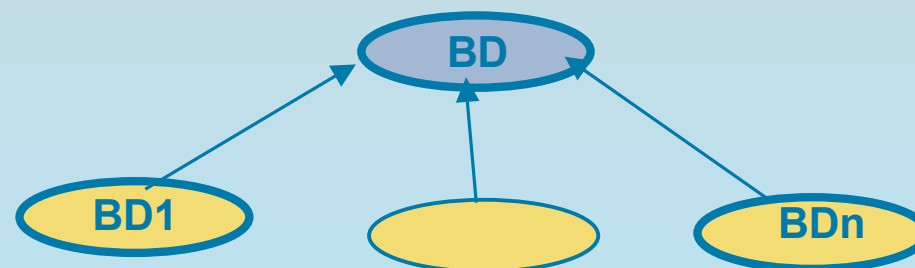


# Migration vers une BDR

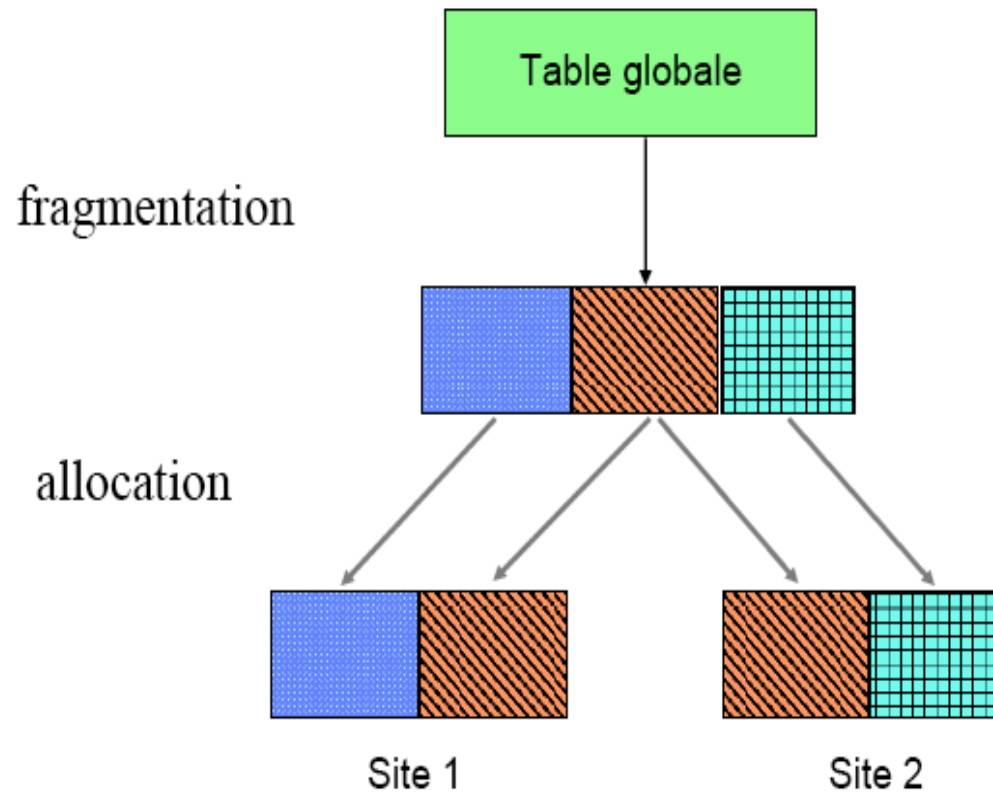
- Décomposition physique en BD locales



- Intégration logique des BD locales existantes

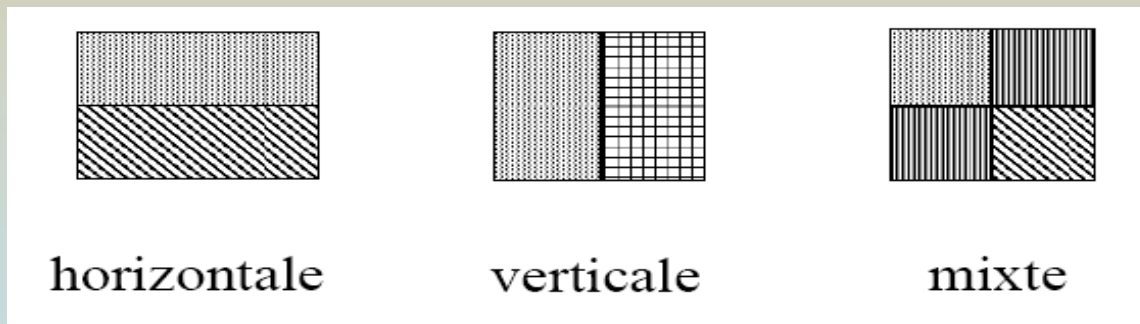


# Conception par décomposition



# Conception par décomposition

- fragmentation
  - trois types : horizontale, verticale, mixte



- performances en favorisant les accès locaux
- équilibrer la charge de travail entre les sites (parallélisme)

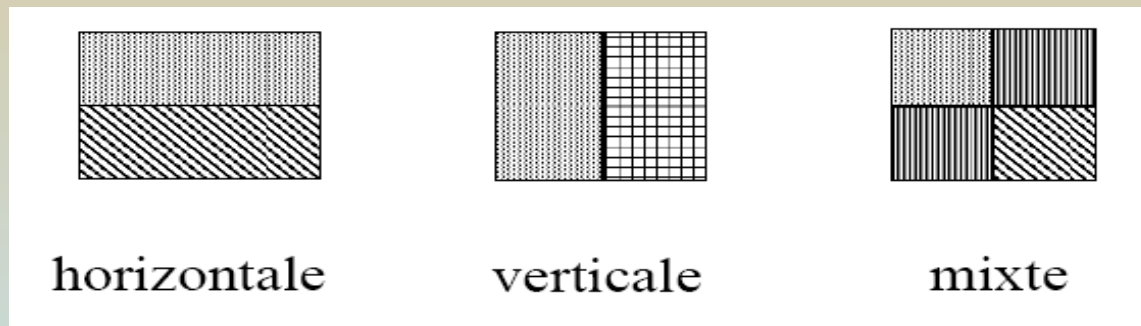
# Conception par décomposition

Fragmentation doit être

- Complète
  - chaque élément de R doit se trouver dans un fragment
- Reconstructible
  - on doit pouvoir recomposer R à partir de ses fragments
- Disjointe
  - chaque élément de R ne doit pas être dupliqué

# Conception par décomposition

- fragmentation
  - trois types : horizontale, verticale, mixte



- performances en favorisant les accès locaux
- équilibrer la charge de travail entre les sites (parallélisme)

- duplication (ou réplication)
  - favoriser les accès locaux
  - augmenter la disponibilité des données

# Conception par décomposition

Fragments définis par sélection

Client1 = Client where ville = 'Paris'

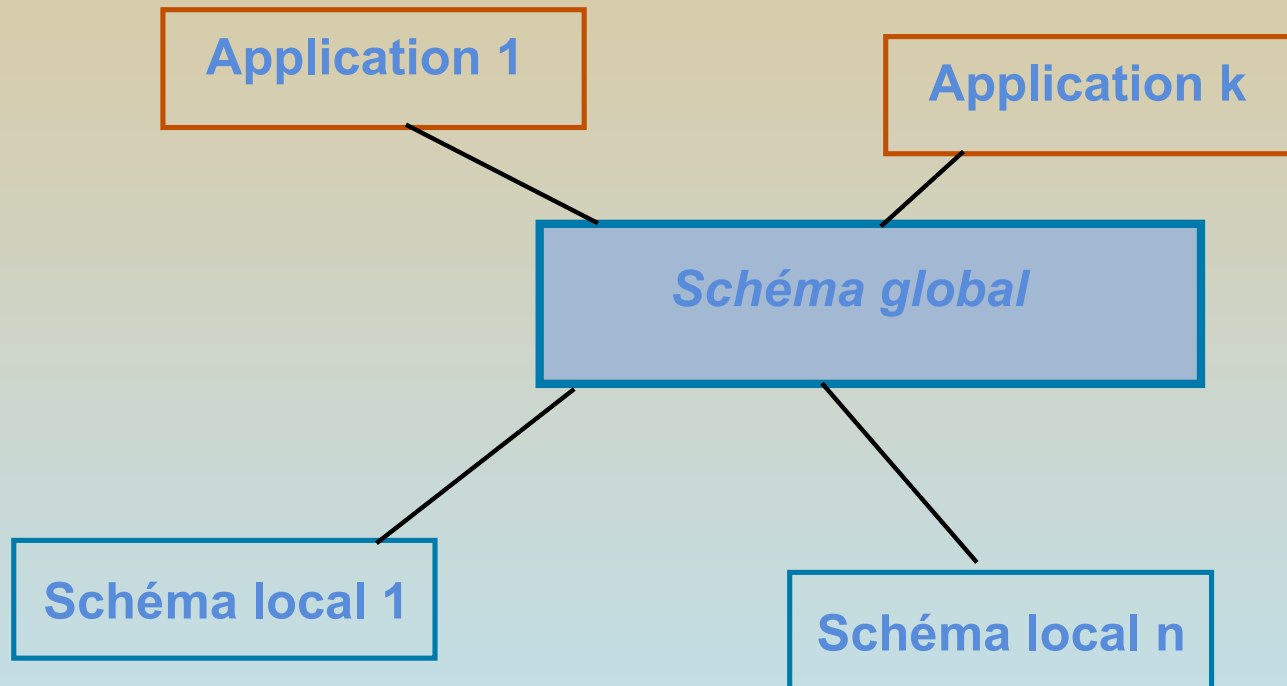
Client2 = Client where ville  $\neq$  'Paris'

client	nom	ville
C1	Dupont	Paris
C2	Martin	Lyon
C3	Martin	Paris
C4	Smith	Lille

Reconstruction

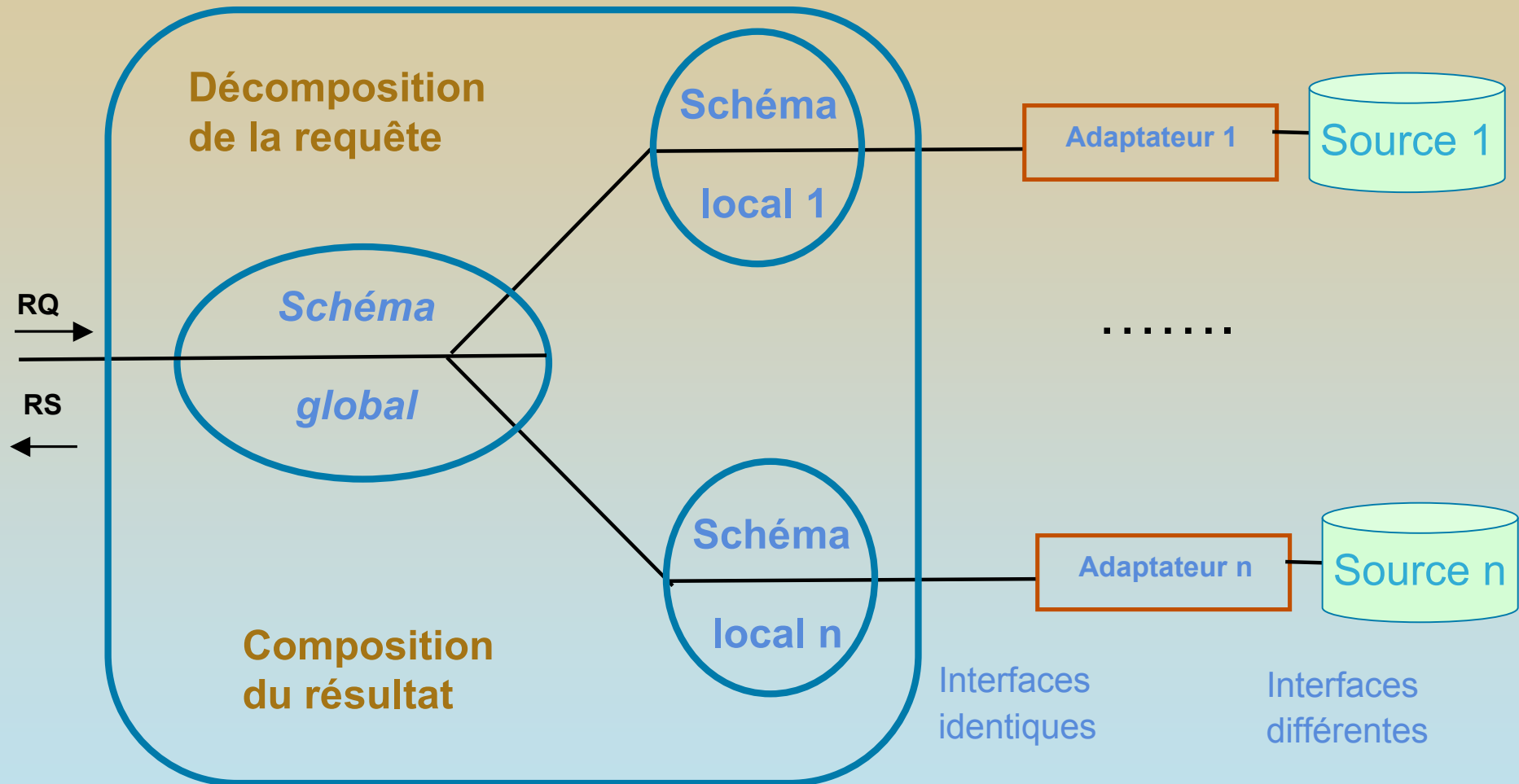
Client = Client1 U Client2

# Architecture de schémas



- Avantage : indépendance applications/BDR
- Inconvénient : schéma global à gérer

# Architecture



- Catalogue global des sources

# Schéma global

## Schéma conceptuel global

Donne la description globale et unifiée de toutes les sources

Offres(emploi, ville, date), Demandes(nom, emploi)

## Schéma de placement

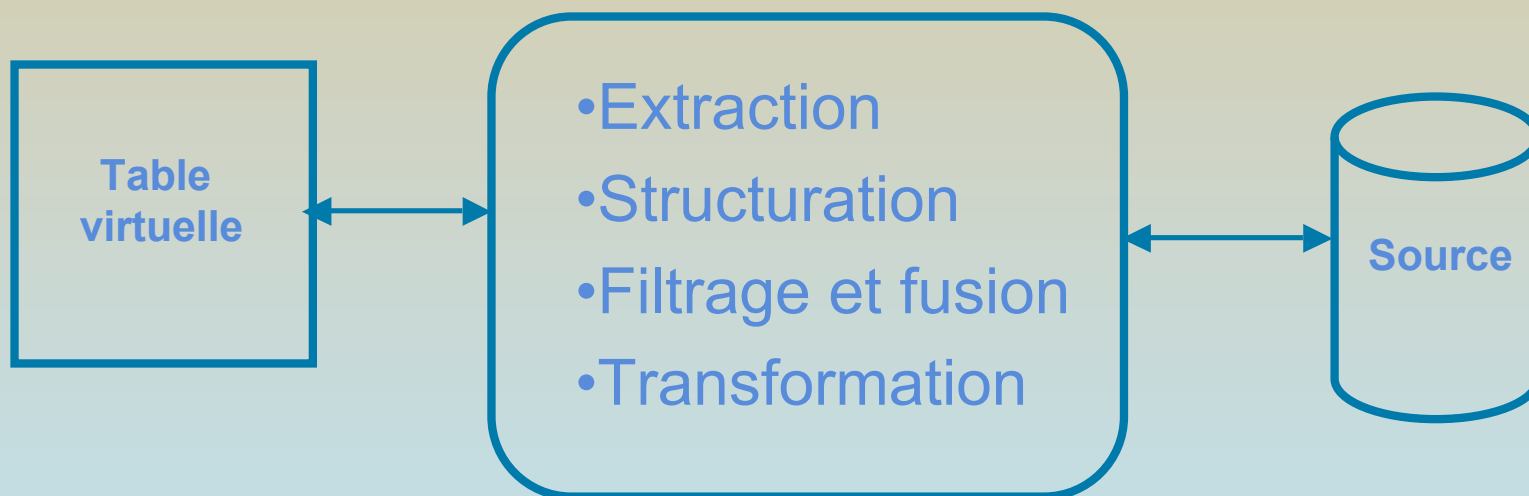
Règles de correspondance avec les données locales

Offres= Offres@site1 ∪ Offres@site2

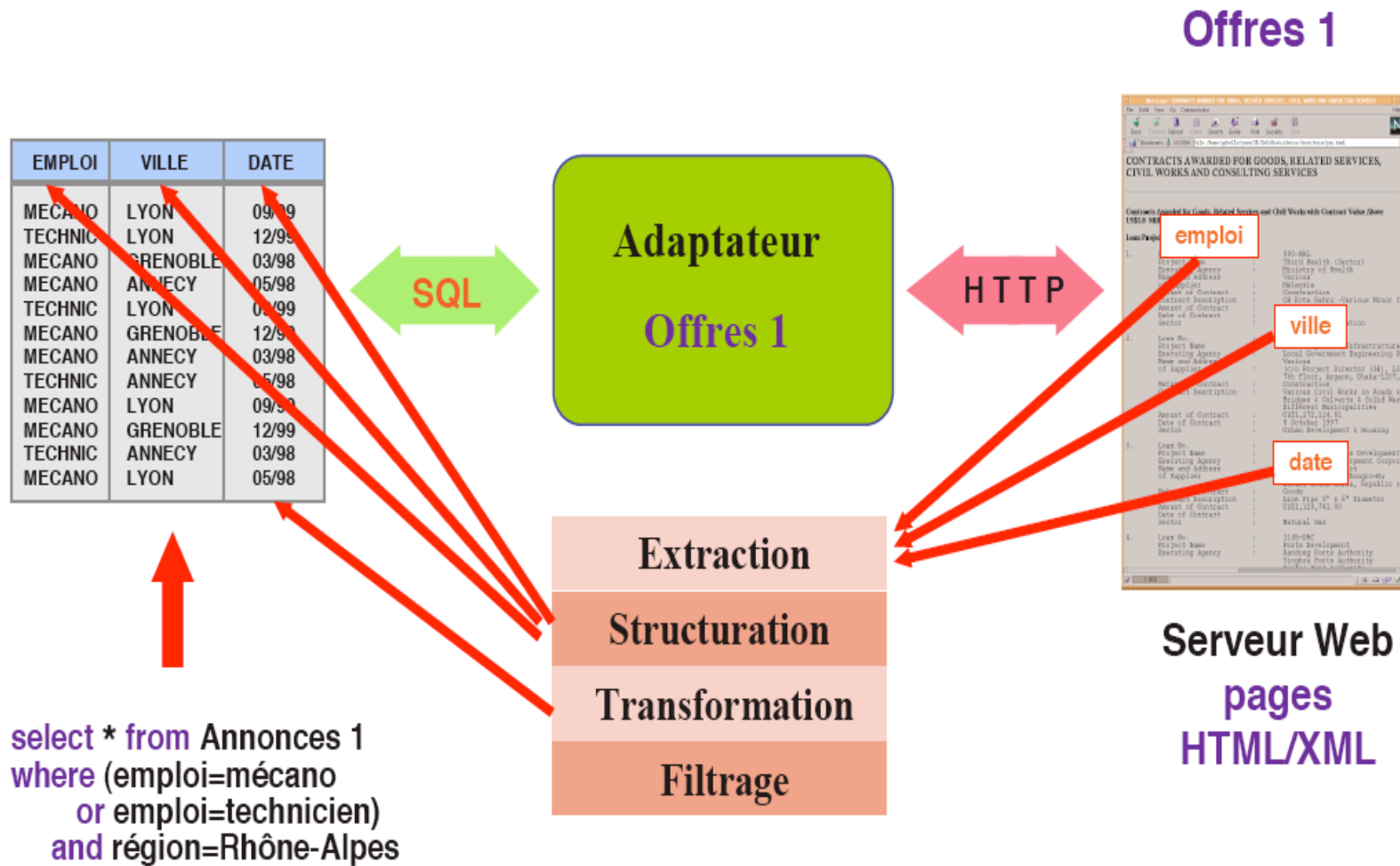
Demandes = Demandes@site1

# Fonctions d'un adaptateur

## Adaptateur



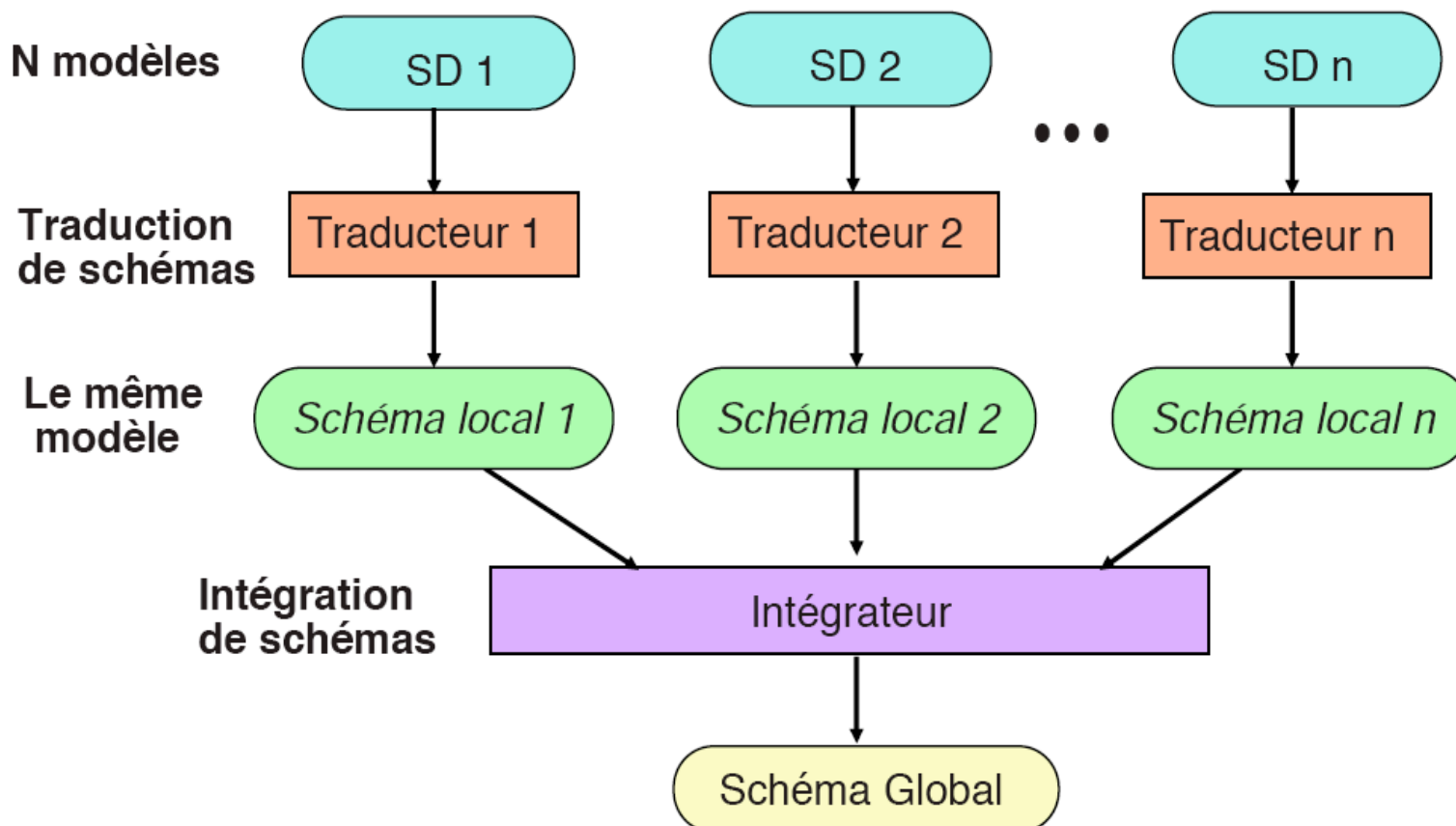
# Exemples d'adaptateur



# Offres d'adaptateurs

- Fournisseurs indépendants
  - extracteurs indépendants entre les données sources et les outils cibles
  - Information Builders Inc. (IBI), Evolutionary Technology Inc. (ETI), Prism, Carleton, etc.
- Editeurs de SGBD
  - passerelles entre le SGBD et les données sources
  - interface standard : ODBC, JDBC, OLE/DB, ADO
  - Oracle, DB2, Microsoft, Sybase, etc.

# Intégration de bases de données



# Modèles d'intégration

- Modèle relationnel
  - structures de données simples et régulières
- Modèle objet
  - structures de données complexes et régulières
- Modèle semi-structuré (XML)
  - structures de données complexes et irrégulières
  - pas de schéma obligatoire

# Intégration de schémas

- 1. pré-intégration
  - identification des éléments reliés et établissement des règles de conversion (e.g. 1 pouce = 2,54 cm)
- 2. comparaison
  - identification des conflits de noms (synonymes et homonymes) et des conflits structurels (types, clés)
- 3. mise en conformité
  - résolution des conflits de noms et des conflits structurels (changements de types, de clés)
- 4. fusion et restructuration
  - fusion des schémas intermédiaires pour créer le schéma intégré

# Intégration en relationnel

Emp = Emp@Site1 U Emp@Site2

prenom	nom	ville	tel.
null	P. Dupont	Paris	0140...
Anne	Martin	Nantes	null
null	A. Martin	Nantes	0235...
Jean	Smith	Lille	null

Emp@Site1

nom	ville	tel.
P. Dupont	Paris	0140...
A. Martin	Nantes	0235...

Emp@Site2

prenom	nomF	ville
Anne	Martin	Nantes
Jean	Smith	Lille

- Problèmes: renommage et introduction de valeurs nulles

# Interrogation en relationnel

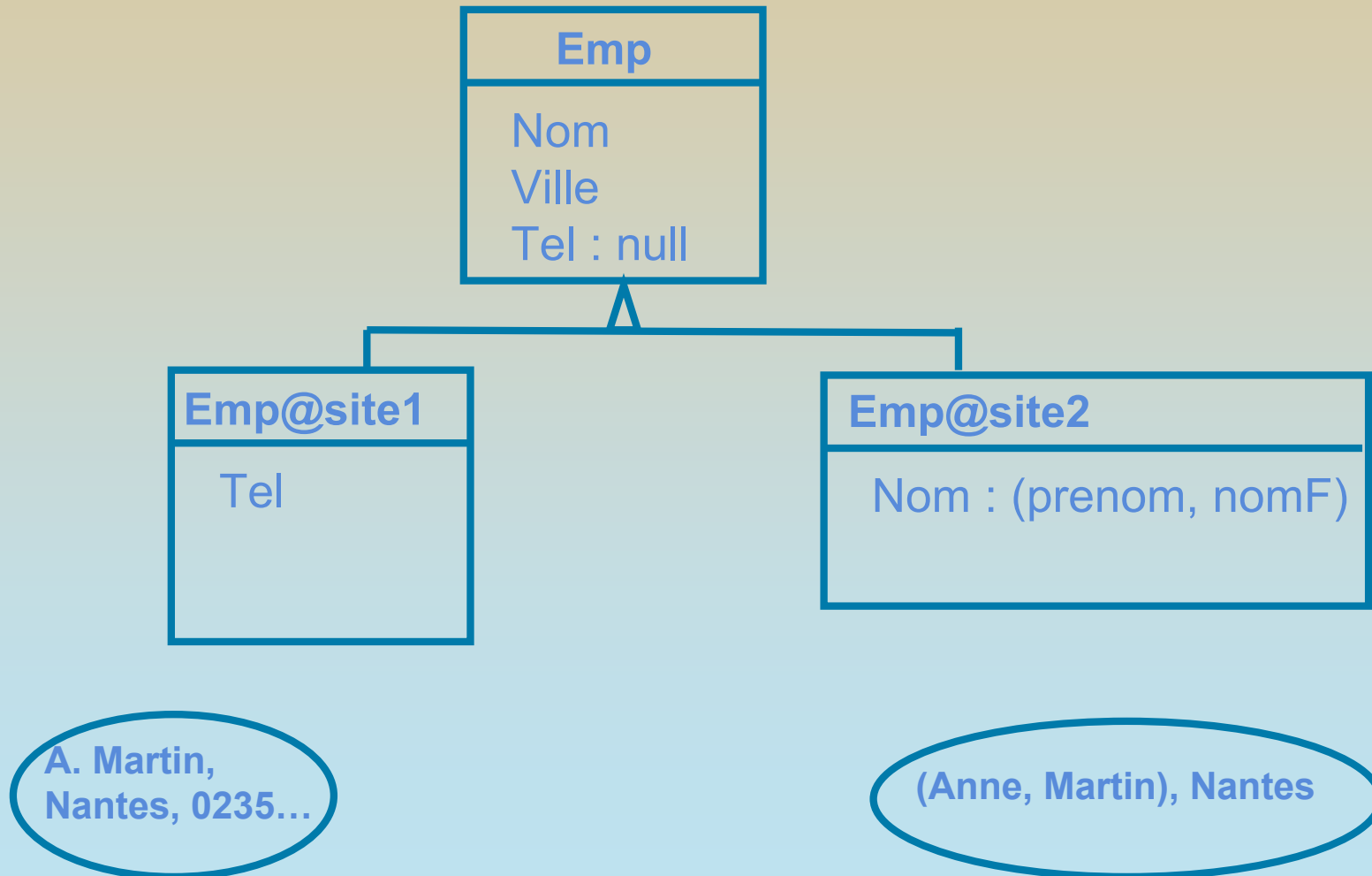
Select prenom, nom, tel  
From EMP  
Where ville=« nantes »

prenom	nom	ville	tel.
null	P. Dupont	Paris	0140...
Anne	Martin	Nantes	null
null	A. Martin	Nantes	0235...
Jean	Smith	Lille	null



prenom	nom	tel.
Anne	Martin	null
null	A. Martin	0235...

# Intégration en objet



# Interrogation en objet

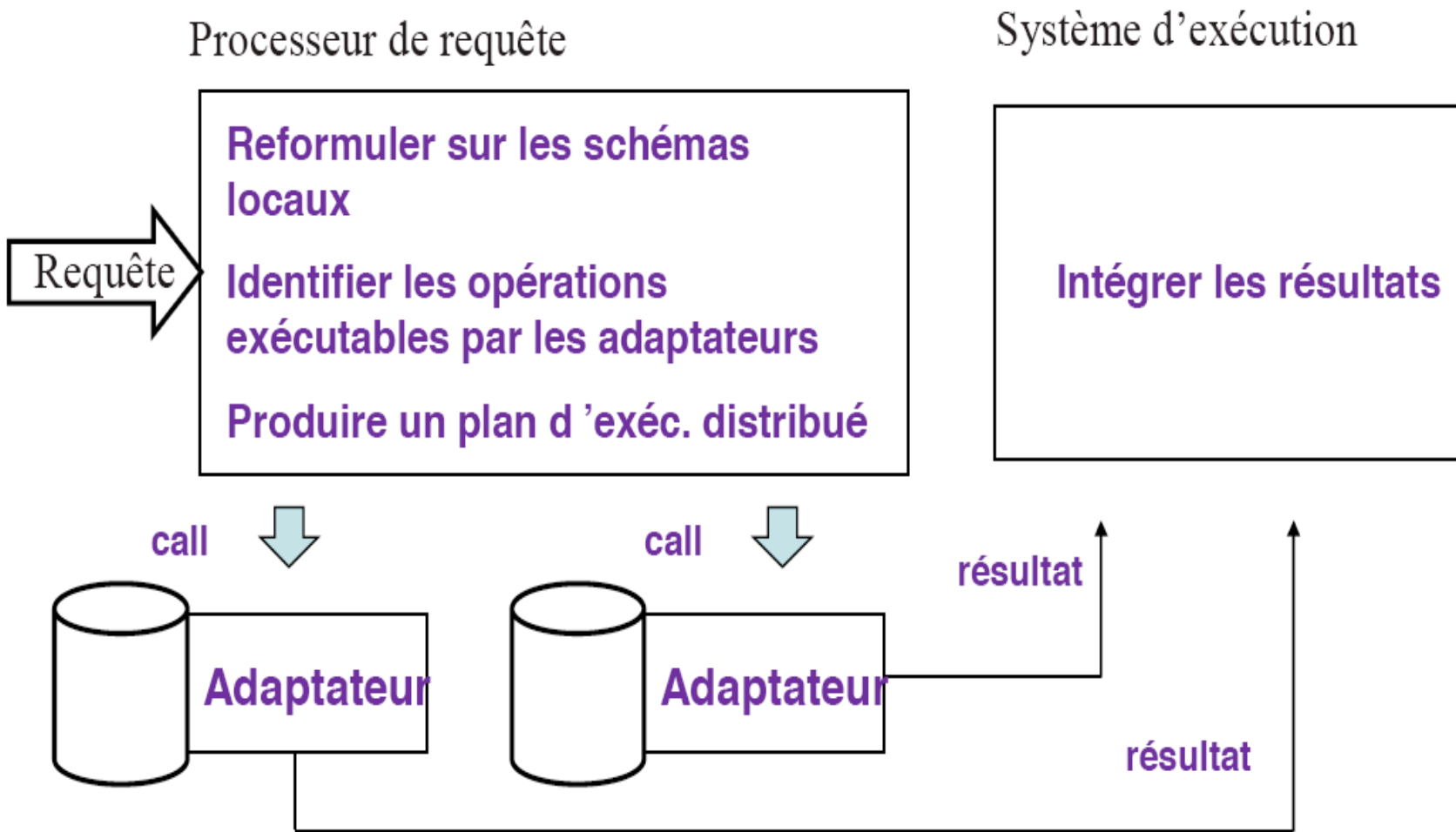
```
Select nom, tel  
From EMP  
Where ville=« Nantes »
```



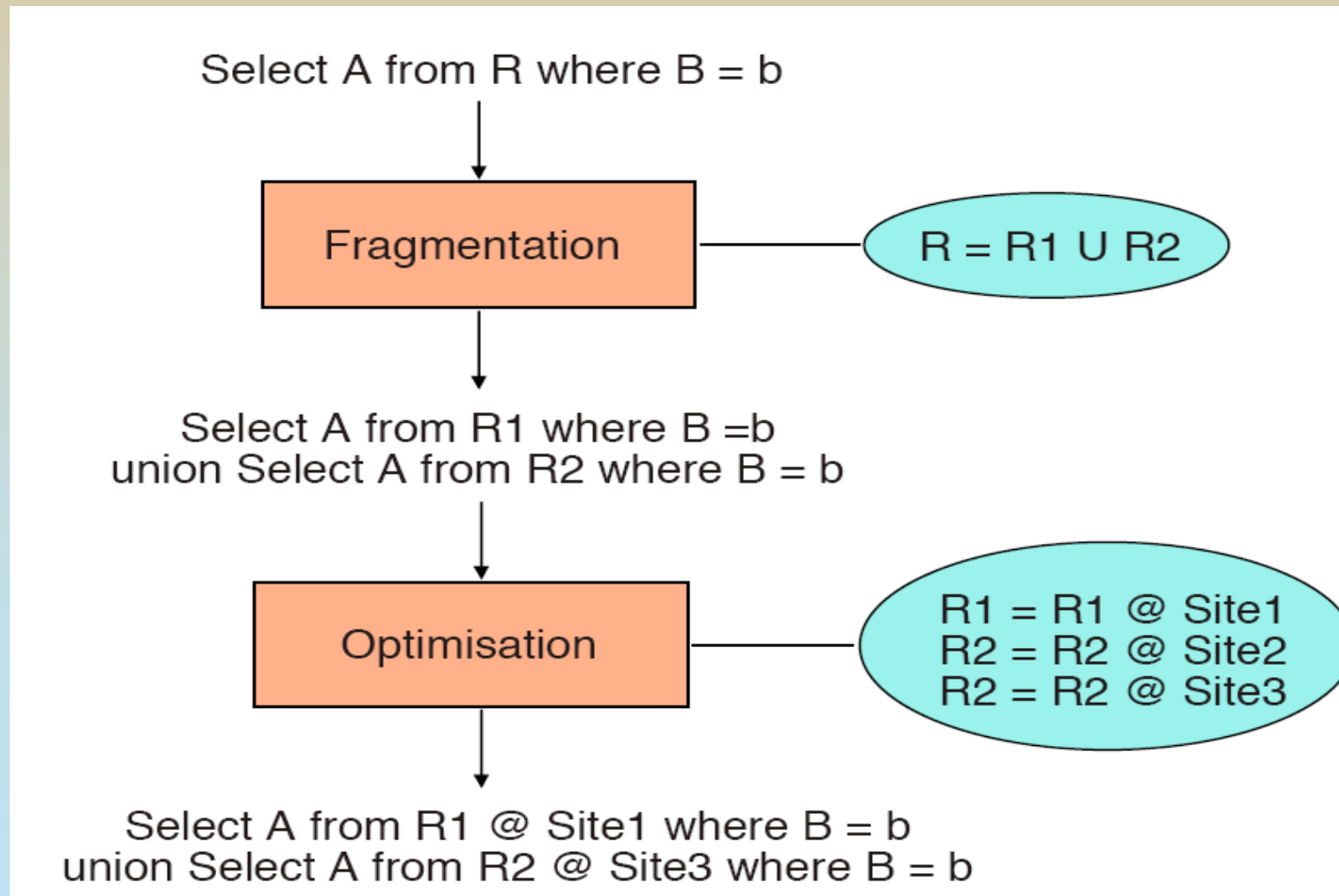
A. Martin, 0235...

(Anne, Martin), null

# Traitement de requête distribuée



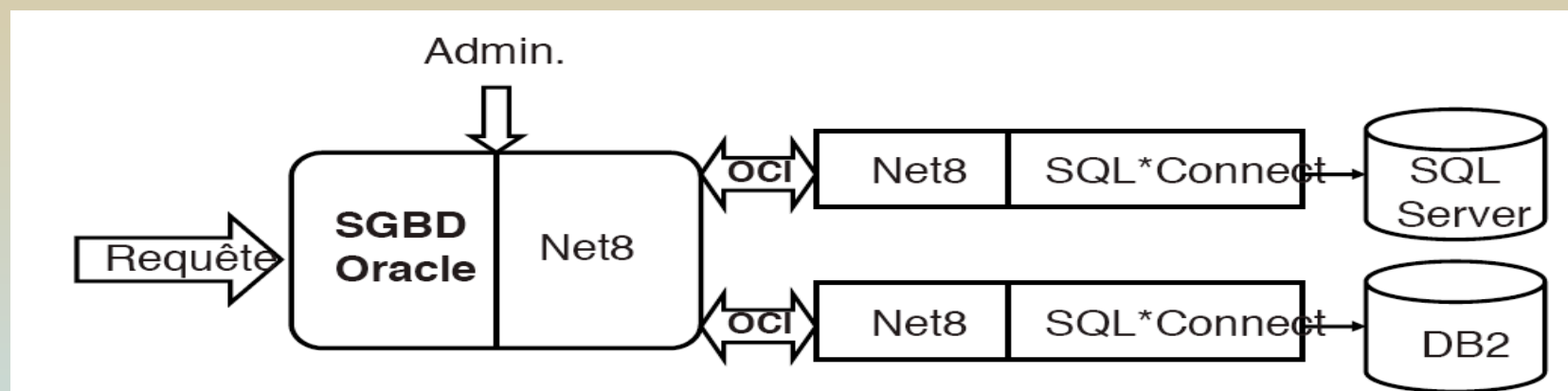
# Exemple de traitement de requête simple



# SGBD distribués

- SGBD relationnels
  - Oracle, Ingres, Sybase, DB2, Informix
- DataJoiner (IBM)
  - basé sur DB2
- VirtualDB (Enterworks)
  - basé sur GemStone, vue objet des tables
- Open Database Exchange (B2Systems)

# Oracle et la distribution des données



- SGBD Oracle
  - gestion du catalogue de la BDR
- SQL\*Net
  - connexion client-serveur, login à distance automatique
  - requêtes distribuées, transactions distribuées, réplication
- SQL\*Connect : passerelle vers les bases non-Oracle

# Database link

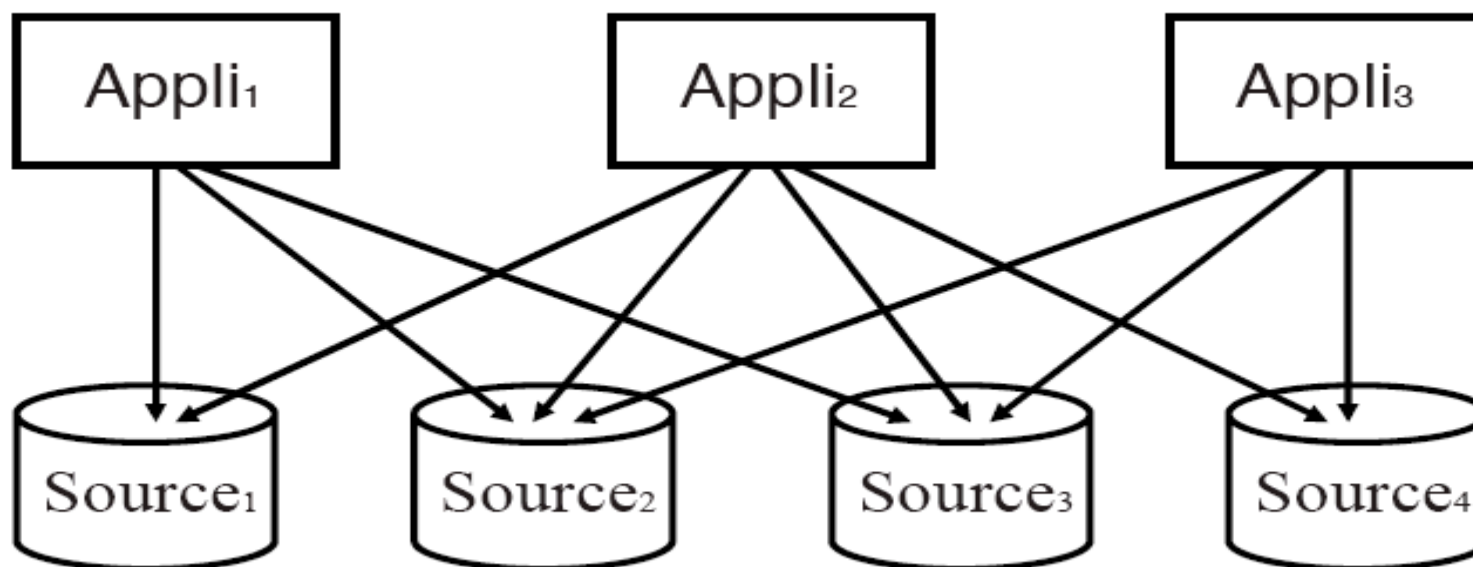
- Lien à une table dans une BD distante spécifié par :
  - nom de lien
  - nom de l'utilisateur et password
  - chaîne de connexion Net8 (protocole réseau, nom de site, options, etc...)
- Exemple

```
CREATE DATABASE LINK empParis  
CONNECT TO anne IDENTIFIEDBY monPW  
USING Paris.emp
```

# Le problème d'intégration d'information

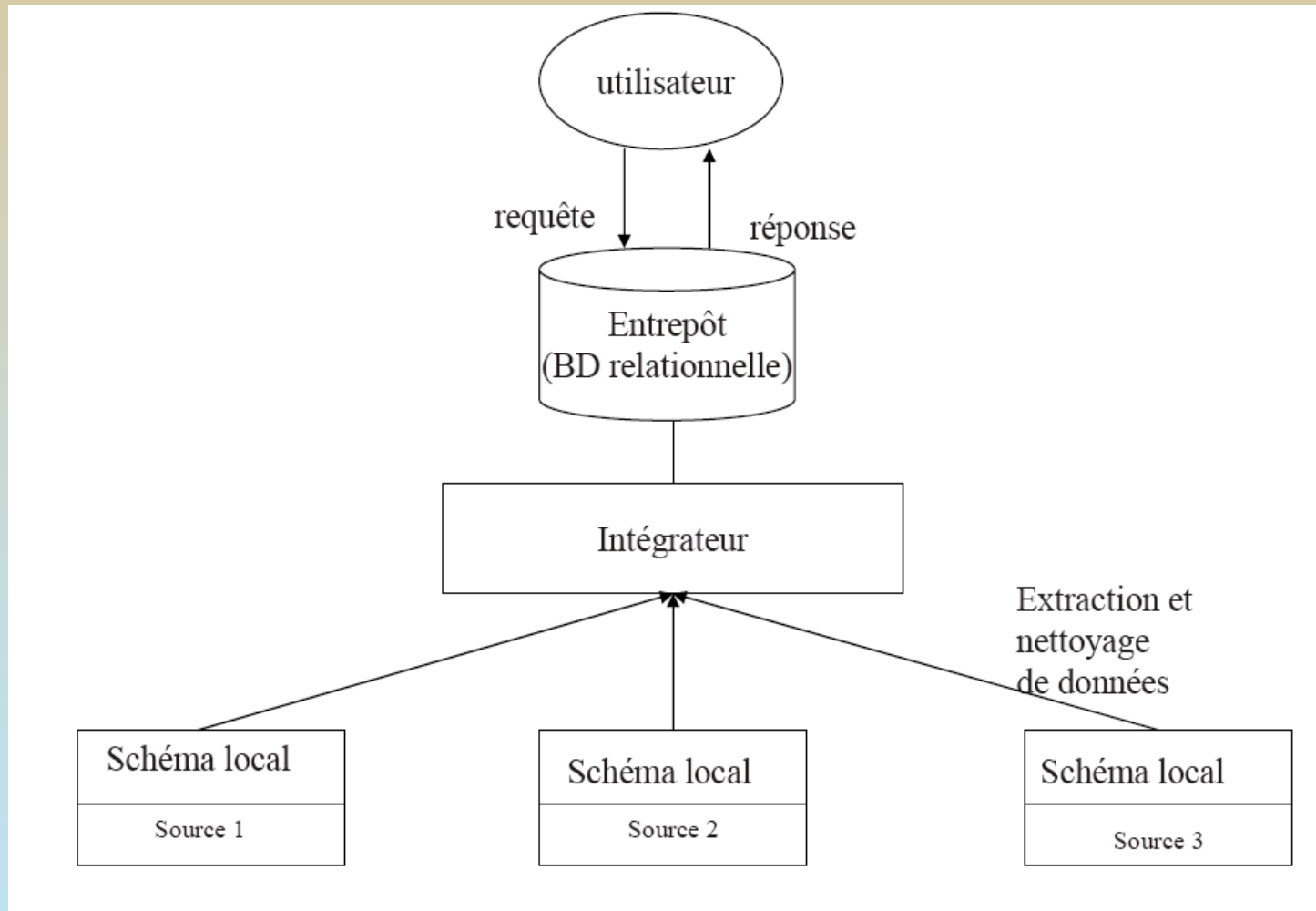
- Prolifération des sources de données distribuées
  - publiques et privées (payantes)
  - très nombreuses
  - autonomes (contrôle local)
  - incohérentes (redondance)
  - hétérogènes
- Famine d'information
  - difficile d'extraire l'information pertinente
  - filtrage manuel
  - coût élevé

# Le problème en client-serveur



- Chaque application doit gérer
  - les communications
  - la manipulation de données
  - les performances
- Ne passe pas à l'échelle

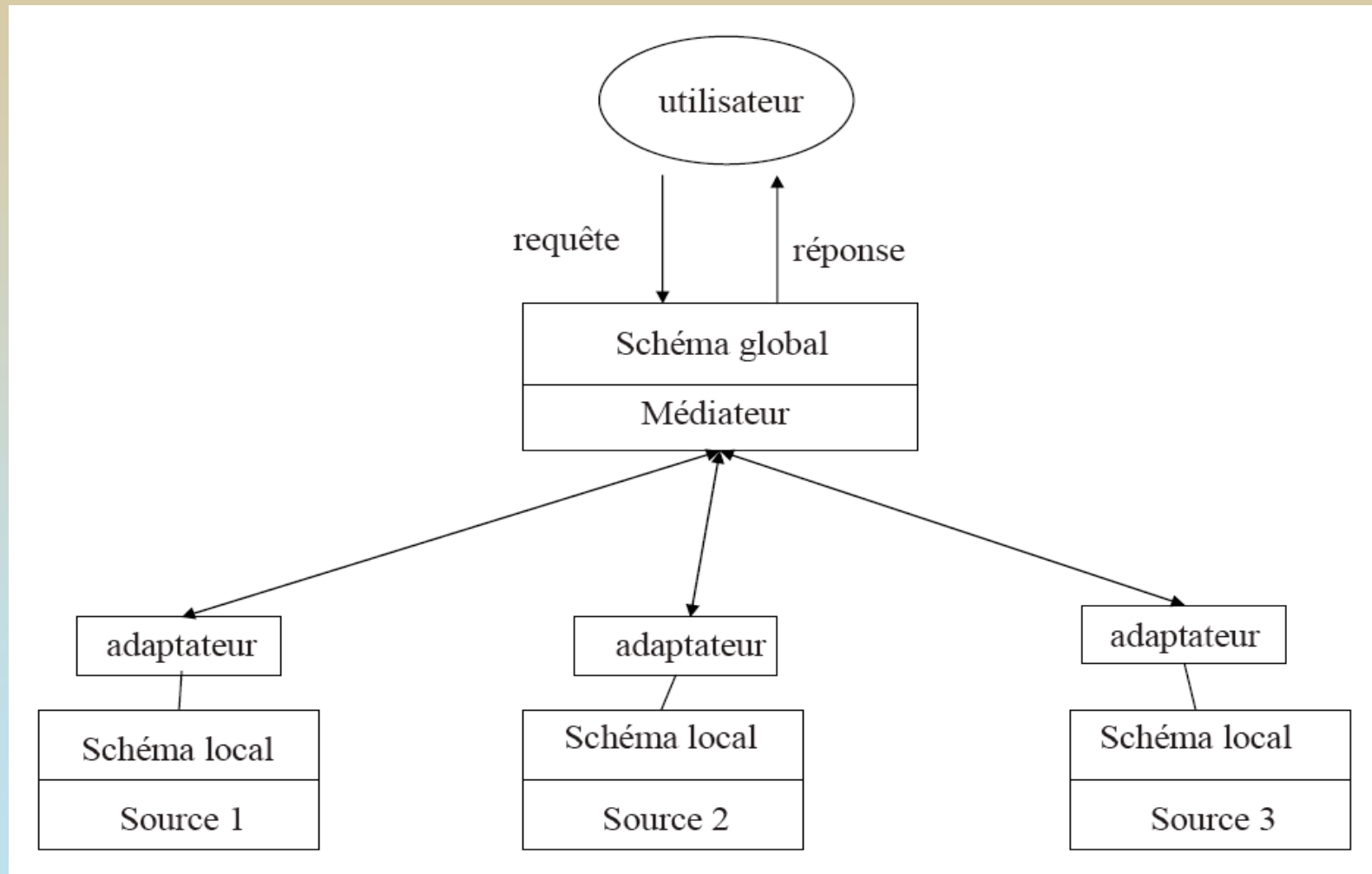
# Entrepôt de données



# Entrepôt de données

- Fonctions
  - Regroupement et récupération de données existantes
  - Référencer les données de manière uniforme
  - Stockage des données et de l'historique des données
  - Mise à disposition des données pour analyse
- Bilan
  - Bonnes performances
  - Données pas toujours fraîches
  - Nettoyage et filtrage des données

# Médiateur

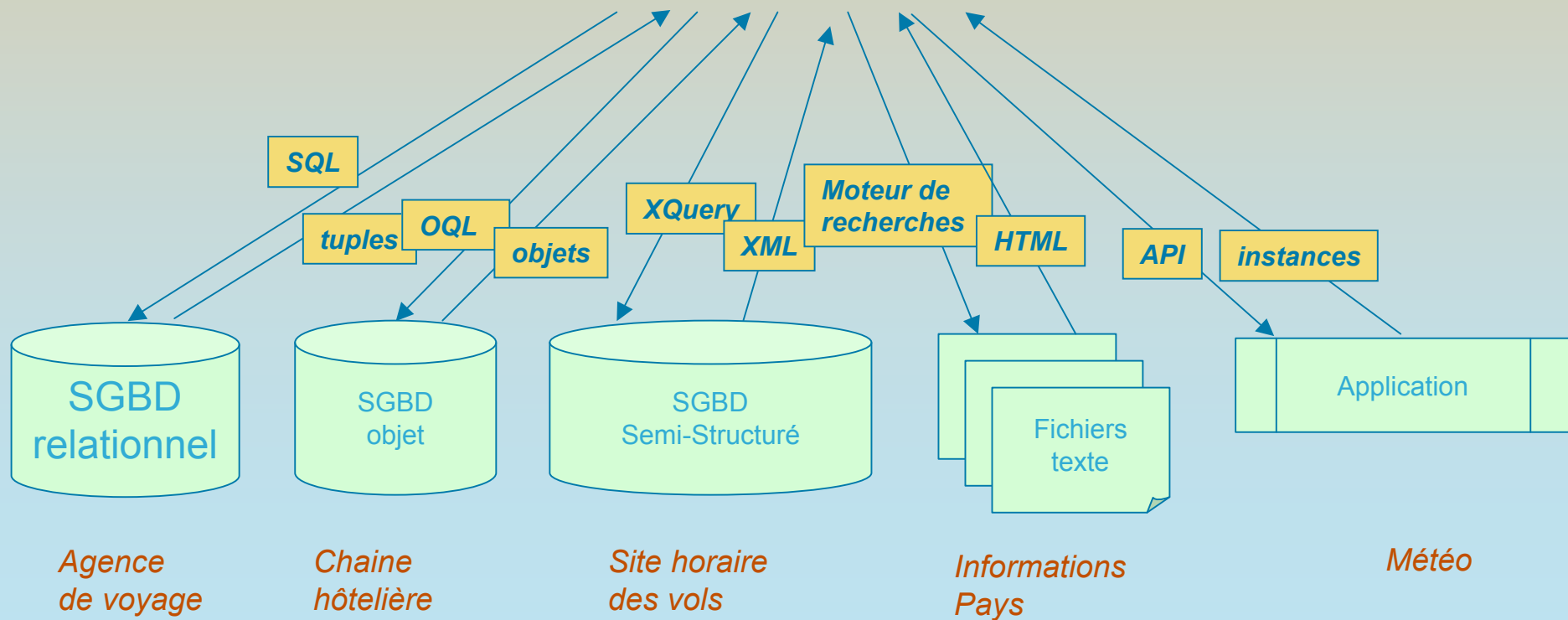
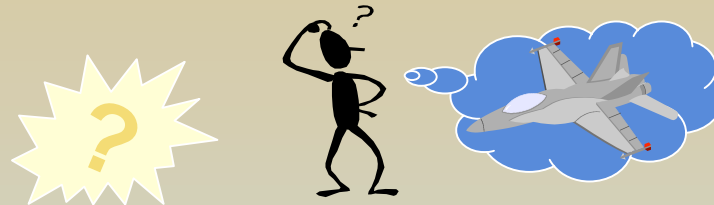


# Médiateur

- Fonctions
  - catalogue global des données
  - intégration de schémas
  - génération d'adaptateurs
  - requêtes distribuées
- Bilan
  - point d'accès unique et uniforme
  - indépendance application/sources => évolution
  - les données sont toujours fraîches
  - traitement de requêtes peut être coûteux
  - performances

# Exemple

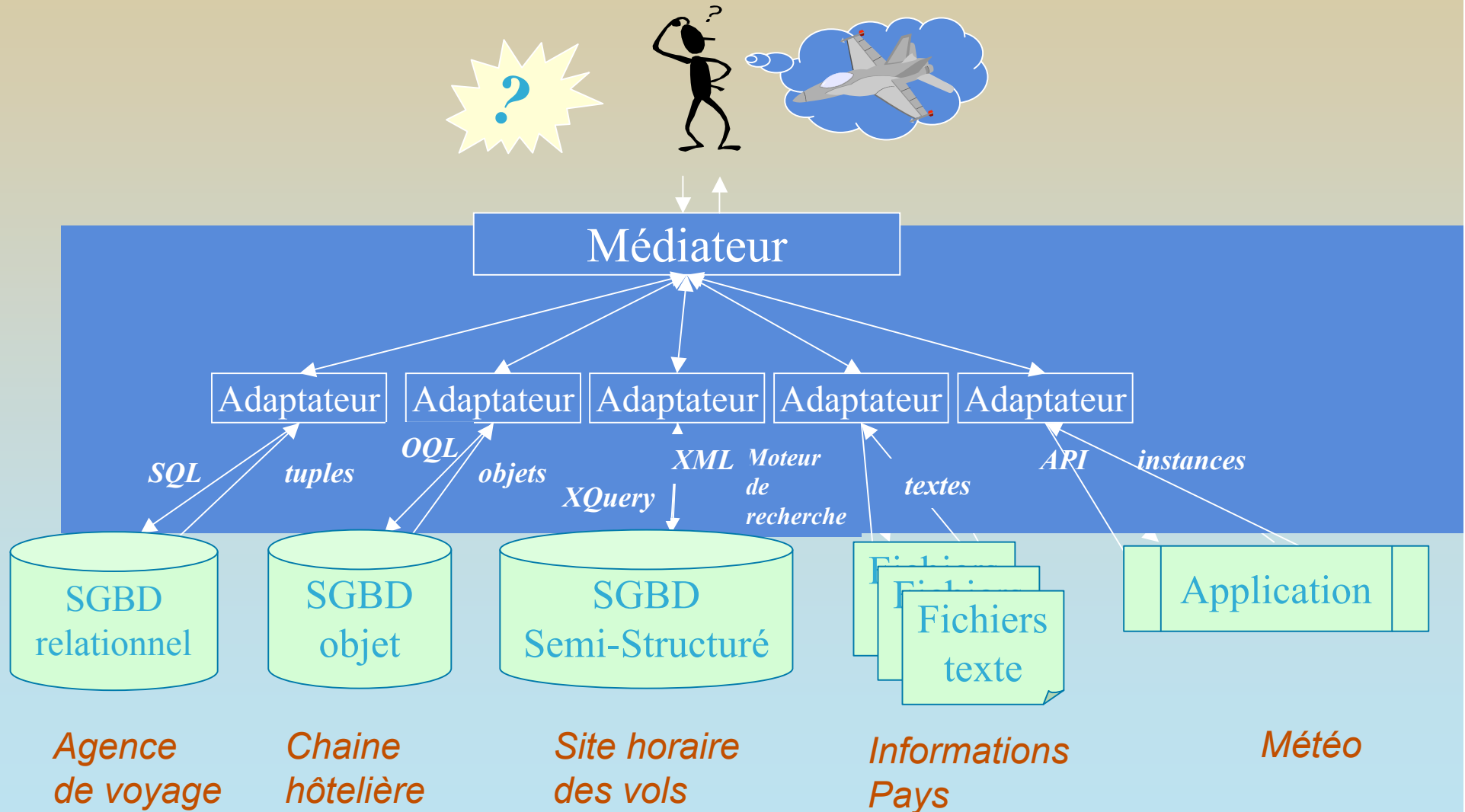
Chercher où passer les vacances



# Systeme interoperable

- Propriétés fondamentales nécessaires à tout système interoperable [Sheth et Larson 1990]:
  - **distribution** : les données gérées par le système proviennent de plusieurs sources.  
Chaque source met une partie de ses données à disposition des autres participants
  - **hétérogénéité** : chaque source choisie et conçue indépendamment des autres (matériel, système d'exploitation, communication, performance, langages, schémas)
  - **autonomie** : une source participant à un système interoperable doit fonctionner comme avant sa participation.

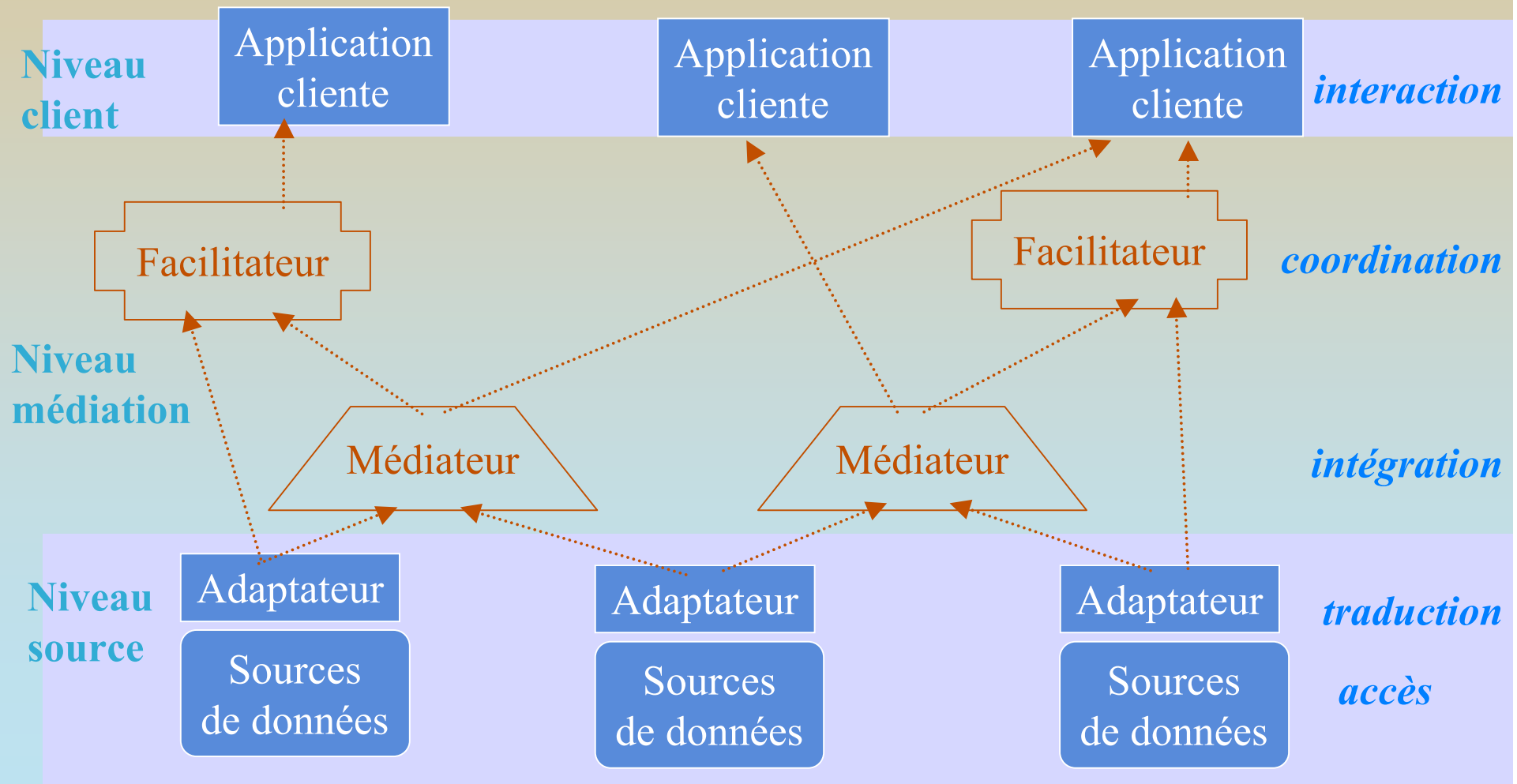
# Architecture de médiation



# Avantages des architectures de médiation

- **accès intégré** par API et portail Web
- **transparence** à la localisation des données pour les applications
- **disponibilité** accrue des données en cas de pannes des serveurs
- support de l'**hétérogénéité** des sources

# Architecture de médiation DARPA I3



# Sources de données

- Une source de données peut être décrite par :
  - localisation
    - référence du site (URL, IP:Port, annuaire LDAP)
    - protocole de communication (TCP/IP, IPX, AppleTalk)
    - moyen d'accès (ODBC, JDBC, API)
    - support (pages Web, SGBD)
  - type de données qu'elle gère  
(structuré (SGBD-R, SGBD-O), semi-structuré (XML, OEM), non-structuré (images, multimédia, textes))
  - possibilité d'interrogation  
(SQL, OQL, propriétaire, moteur de recherche web)
  - format des résultats  
(XML, HTML, ResultSet (tuples), OEM, textes)

# Communication médiateur/adaptateur

- Pour faciliter le travail d'intégration, on définit
  - un langage commun dans lequel le médiateur interrogera les adaptateurs
  - un format de résultat commun dans lequel les adaptateurs répondront au médiateur.
- Ce langage et format de résultat communs peuvent être propriétaires ou standardisés

# Adaptateur (Wrapper)

- L'adaptateur (Wrapper) s'occupe de l'hétérogénéité des sources. C'est un "traducteur".
  - Traduction du langage de requête commun en langage de requête natif (propre à la source)
  - Traduction des résultats natifs en résultats au format commun

# Médiateur

- Le **médiateur** s'occupe de la distribution des sources
  - Localisation des sources
  - Décomposition des requêtes en requête adaptée pour chacune des sources
  - Envoi des requêtes aux sources
  - Recomposition des différents résultats provenant de chacune des sources

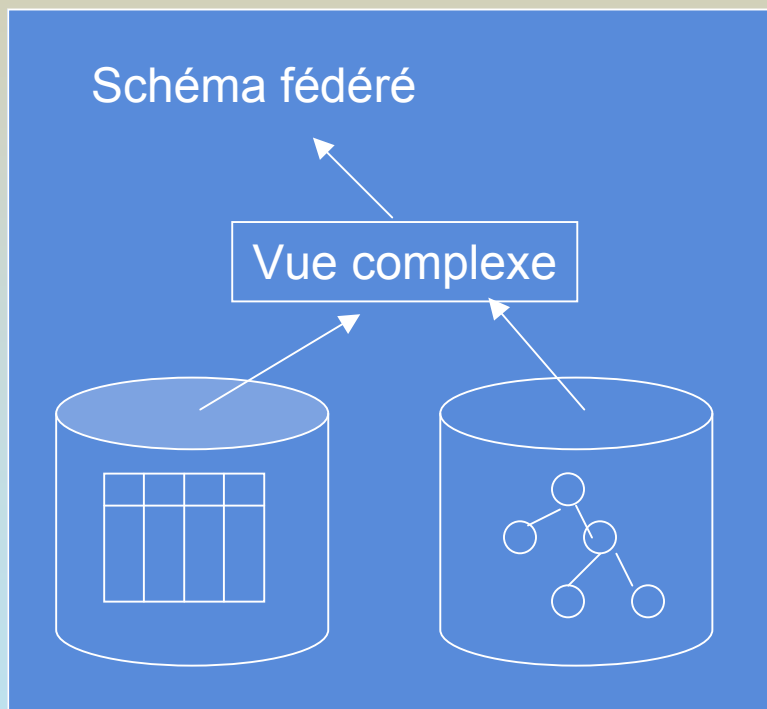
# Intégration des schémas

- Comporte différents aspects
  - Comparaison de schéma
  - Unification de schéma
  - Fusion de schéma
- Difficultés
  - Conflit de niveau d'abstraction
  - Conflit de définition de classes
  - Conflit de divergence schématique

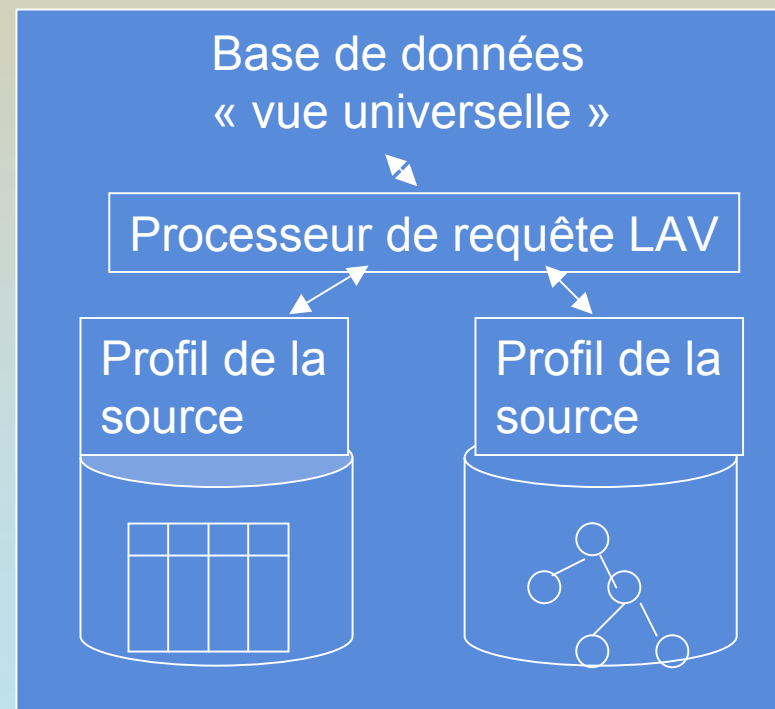
# Architecture GAV et LAV

- GAV : Global as View
  - Schéma global défini comme une vue intégrante sur schémas locaux
  - Approche ascendante depuis les sources vers le médiateur
- LAV : Local As View
  - Chaque source locale est définie comme une vue locale du schéma global
  - Approche descendante depuis le médiateur vers les sources

# Décomposition versus Recomposition



GAV



LAV

# Traitement des requêtes

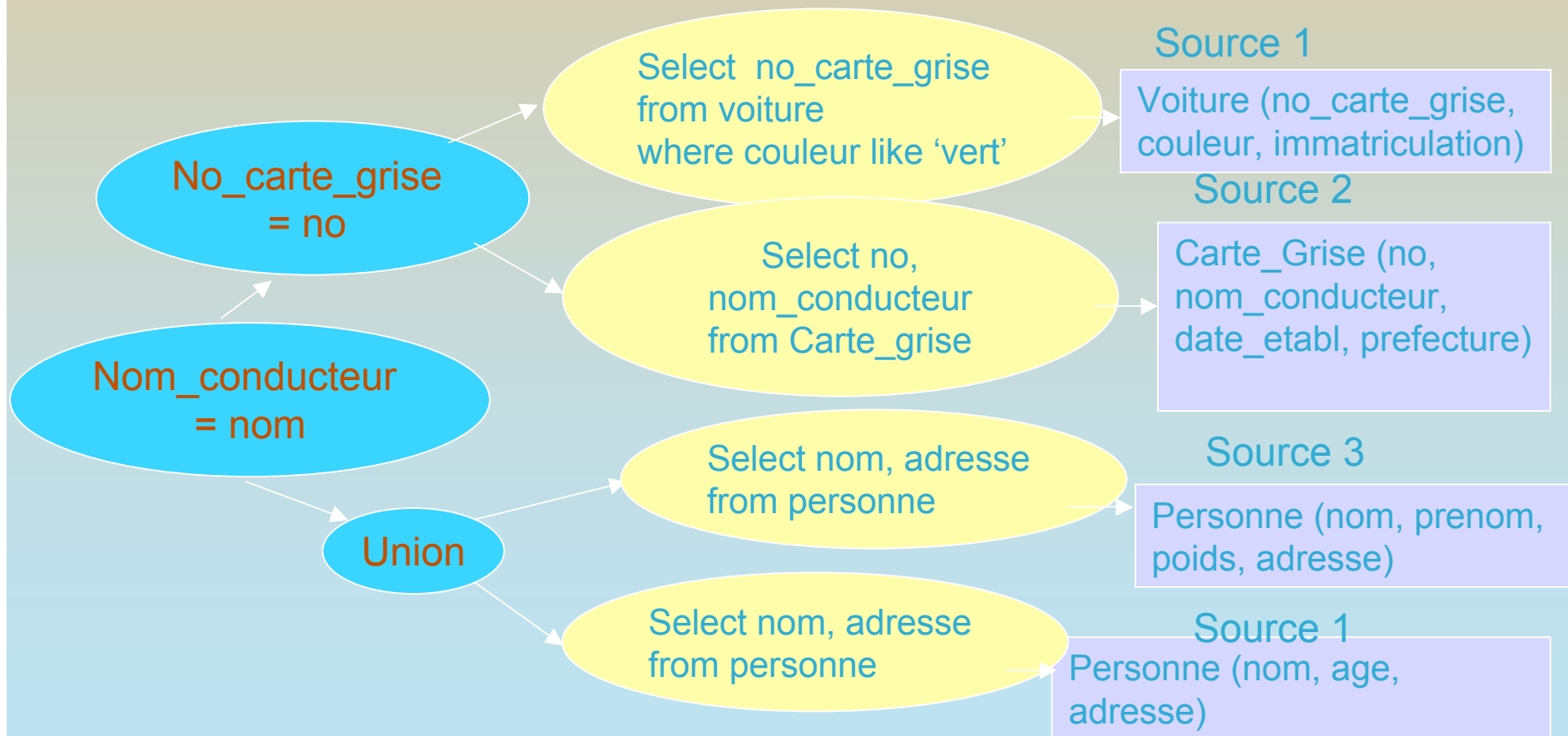
- Analyse syntaxique et sémantique
- Décomposition des requêtes
- Exécution des requêtes sur les sources
  - transformation de la requête en langage commun vers le langage de la source
  - transformation du résultat au format de la source vers le format commun
- Recomposition des résultats
  - combinaison des résultats locaux
  - requêtes de recombinaison sur système global ou un des sites composants.

# Plans d'exécution

- Un plan d'exécution décrit la méthode d'exécution d'une requête. Il est souvent représenté par un arbre algébrique.
- Un arbre algébrique est un arbre où les nœuds sont des opérateurs algébriques et les feuilles les sources de données.
- Il peut exister plusieurs voire une infinité de façon d'exécuter une requête (toutes représentée par des plans d'exécution équivalents). L'ensemble des plans d'exécution permettant de résoudre une requête est appelé espace de recherche.

# Décomposition des requêtes

- Exemple : chercher l'adresse de tous les propriétaires de voiture verte



# Puissance d'interrogation des sources

- Toutes les sources n'ont pas **les mêmes possibilités** d'interrogation
  - SGBD : possibilité de requêtes souvent complexes
  - Moteur de recherche : par mots-clefs
  - Fichiers : via champ indexé
- Le médiateur ou l'adaptateur de la source doit pallier les déficiences de la source
  - si adaptateur : implémentation complexe de chaque adaptateur, intégration simple au niveau médiateur
  - si médiateur : implémentation simple des adaptateurs, intégration complexe au niveau médiateur. Nécessite une communication des capacités de la source au médiateur.

# Optimisation des requêtes

- Stratégies classiques de remontée de projection, restriction, etc.
- Ordonnancement des jointures
- Stratégie sur les jointures inter-sites
  - par interrogation multiples d'une source avec les résultats du premier
  - par boucles imbriqués
  - par tri fusion

# Optimisation statique de requêtes hétérogènes

- Ajout de vues transitoires
- Décomposition d'une requête
- Simplification d'une requête
- Prise en compte des capacités réduites des sources
- Parallélisation d'une requête
- →Élaboration d'un plan optimisé

# Optimisation dynamique de requêtes hétérogènes

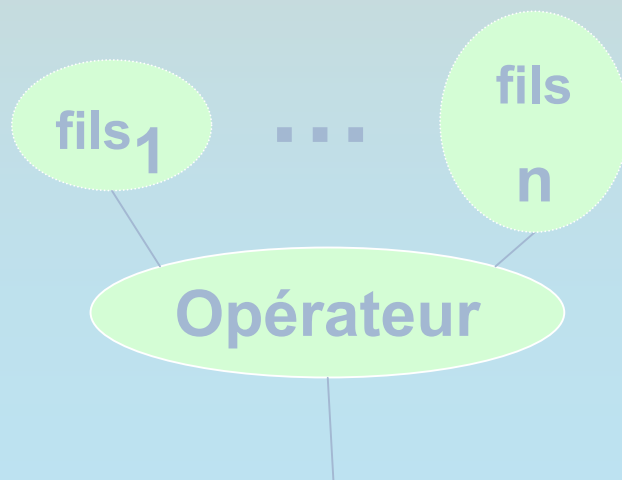
- Reformulation dynamique du plan d'exécution
- Prise en compte de sources indisponibles
- Ordonnancement dynamique des jointures
- Optimisation adaptative

# Modèles de coûts

- Un **modèle de coût** permet d'estimer le coût que prendra un plan d'exécution.
- But : choisir parmi tous les plans d'exécution celui de coût minimal pour l'exécution
- S'appuie sur les **statistiques** et des formules de coûts.
- Statistiques :
  - Du système : Système d'exploitation (CPU, E/S), SGBD (taille d'une page, taille cache)
  - Des données : cardinalité d'une collection, sélectivité d'un attribut, etc.

# Modèle de coût au niveau médiateur

Coût d'exécution d'une requête  
= coût\_communication  
+ opération\_médiateur  
+ coûts\_sur\_les\_adaptateurs  
+ congestion\_du\_réseau



Dépend du débit, de la taille des données à transférer

Formule classiques de coût de calcul d'opérateurs en mémoire

$coûts\_fils \in [max (coût\_fils_j), \Sigma(coût\_fils_j)]$   
suivant degré de parallélisme

Difficile à gérer : latence et temps d'attente au moment de l'exécution de la requête

# Coût sur les adaptateurs

- Les sources sont indépendantes et ne communiquent pas forcément leurs informations de coûts.
- Différentes stratégies permettant d'estimer le coût d'une requête sur une source
  - Estimation analytique
    - Soumission de formules
  - Apprentissage progressif
    - Gestion d'historique

# Modèle de coût

## *Coût d'une architecture de médiation*

- **Calibration [PEGASUS]**
  - requêtes types pour calibrer paramètres de la source
  - affinée avec échantillonnage
  - pour données objets [IRO-DB]
- **Historique [HERMES]**
  - s'appuie sur les statistiques des requêtes précédentes
- **Défini par les adaptateurs [GARLIC]**
  - modèle de coût défini séparément pour chaque adaptateur
- **Générique [DISCO]**
  - intégrer modèle de coût des adaptateurs + hiérarchie de coût et coût par défaut pour coût manquant d'un adaptateur

## *Coût sur données semi-structurées*

- **Coût sur modèle semi-structuré dans un entrepôt [LORE]**

# Gestion de Cache

- **Cache de pages** : adapté à des SGBD classiques, peu adapté aux autres sources (Web, ou opaques)
- **Cache de tuples** : faisable pour les pages web (proxy). Mais difficile de préciser quels sont les tuples déjà dans le cache.
- **Cache sémantique** : Garder un historique des prédicats de requêtes déjà posées.
  - requête dans le cache local
  - requête complémentaire
  - actualiser le cache

# Médiateur existants

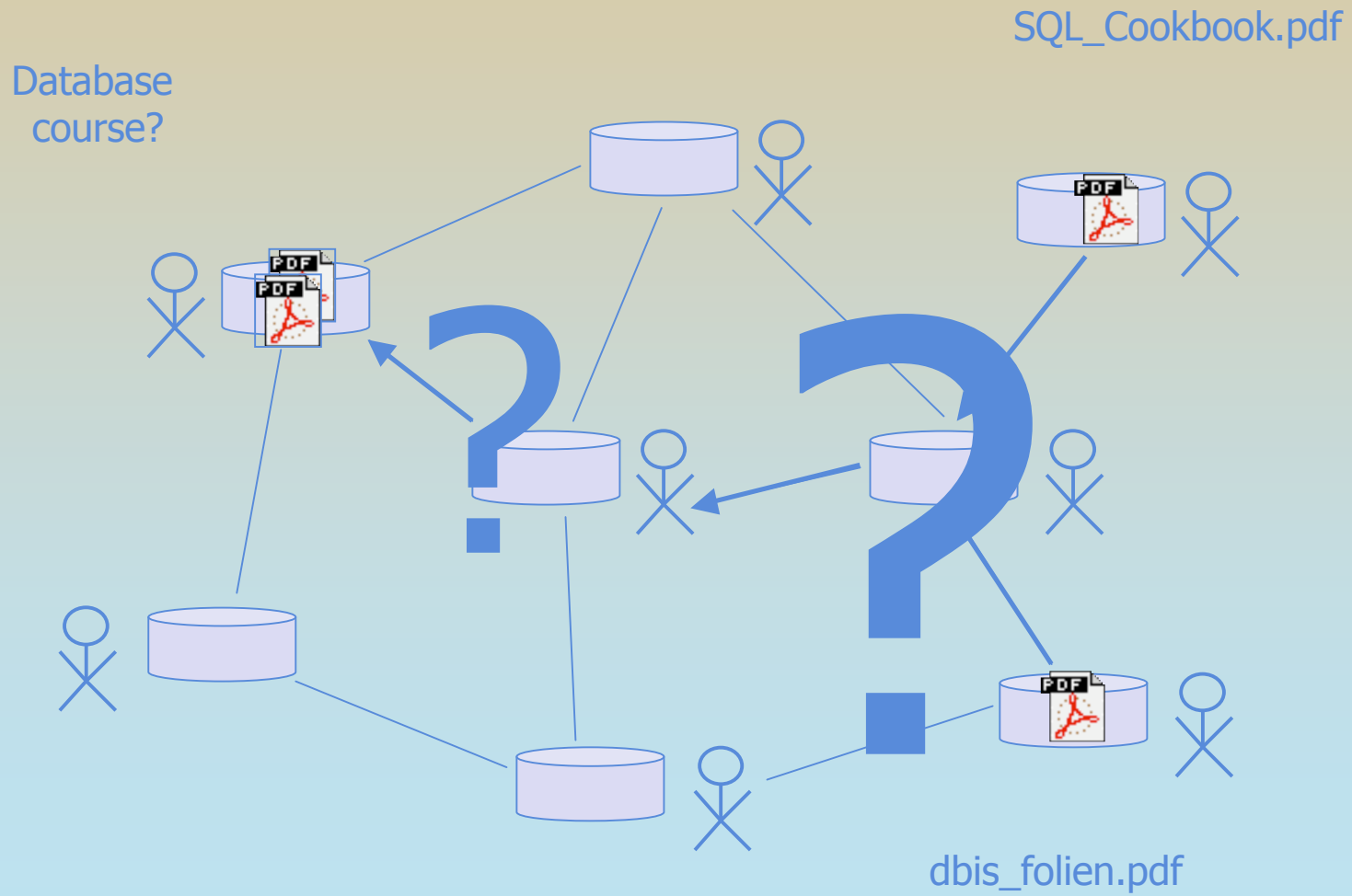
- Génération relationnelle (1975-1990)
  - Souvent centré sur un SGBD qui joue le rôle d'un médiateur
  - SDD1, Sirius Delta, R\*, Ingres/Star
  - Mermaid, Multibase, MSQ
- Génération relationnelle étendue (1990-2000)
  - Fédère des BD hétérogènes autour de SQL3
  - Objet : OLE-DB, pegasus, IRO-DB
  - XML : Medience Server, Information Integrator (IBM)
- Génération XML XQuery (2000- ...)
  - OLE-DB.NET (Microsoft), Nimble, Xquark Fusion,
  - Liquid Data (BEA), Enosys Software

# Réseau P2P

Réseau d'ordinateurs, où chaque noeud est à la fois client et serveur : met à disposition des ressources, et accède à des données se trouvant sur d'autres noeuds.

- Pas de vision centralisée du système
- Pas de coordination centralisée
- Pas de schéma
- Principe de propagation des requêtes
- Grande autonomie des noeuds
- Accès à des fichiers

# Réseau P2P



# Conclusion

- Internet s'étend
  - Sources d'information de plus en plus nombreuses
  - Informations de plus en plus hétérogènes
- Médiation de plus en plus nécessaire
  - base de connaissances
  - portails d'information
  - moteur de recherche spécifiques

# Comparaison

	Modèle	Requêtes	Admin.	Hétérogénéité	Autonomie	Nb de sources
SGBD répartis	Relationnel	Requêtes complexes, transactions	Globale	Faible	Faible	dizaines
Entrepôts	Relationnel	Complexes, lecture	Globale	Faible	Forte	dizaines
Médiateurs	Relationnel, objet, semi-structuré	Simple, textuelles	Locale	Forte	Forte	centaines
P2P	Fichiers	Simple	Locale	Forte	Forte	milliers

# Bibliographie

- A. Halevy, Answering queries using views : a survey, VLDB Journal, 2001.
- <http://www.cs.washington.edu/homes/alon/>
- S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J.D.Ullman, J. Widom : The TSIMMIS Project: Integration of heterogeneous information sources, in 16th Meeting of the Information Processing Society of Japan, Tokyo, 1994.
- I. Manolescu, D. Florescu, D. Kossman : Answering XML queries over heterogeneous data sources, BDA 2001.
- G. Wiederhold: Mediators in the architecture of future information systems, Computer, 25(3):38-49, 1992.
- G. Wiederhold: Mediation in information systems. ACM Computing Surveys, 27(2), 1995.