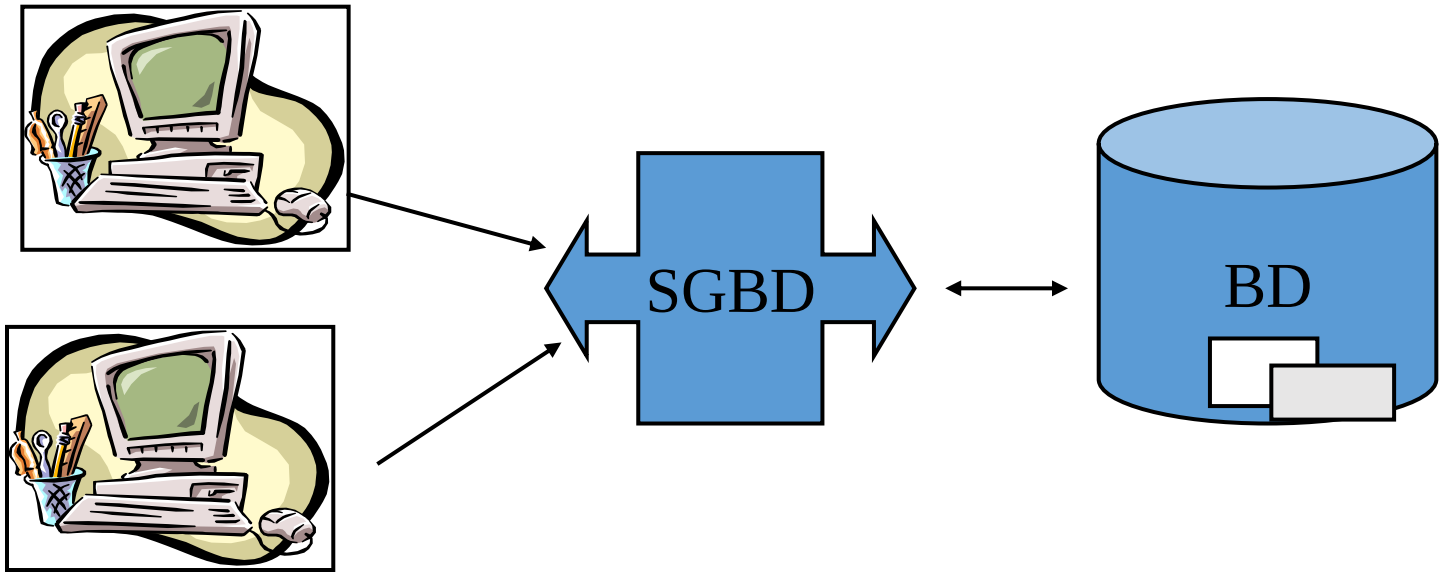
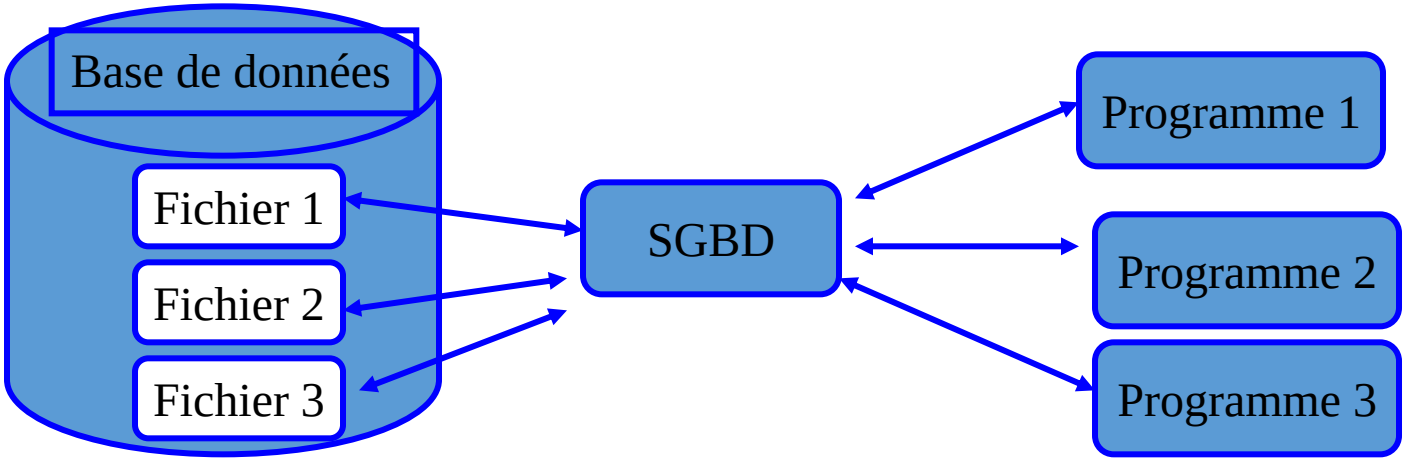
CHAPITRE N° I
INTRODUCTION À L'ARCHITECTURE DES SGBD

© Pr. Habib Ounalli
Département d'Informatique
Faculté des Sciences de Tunis
habib.ounelli@fst.rnu.tn

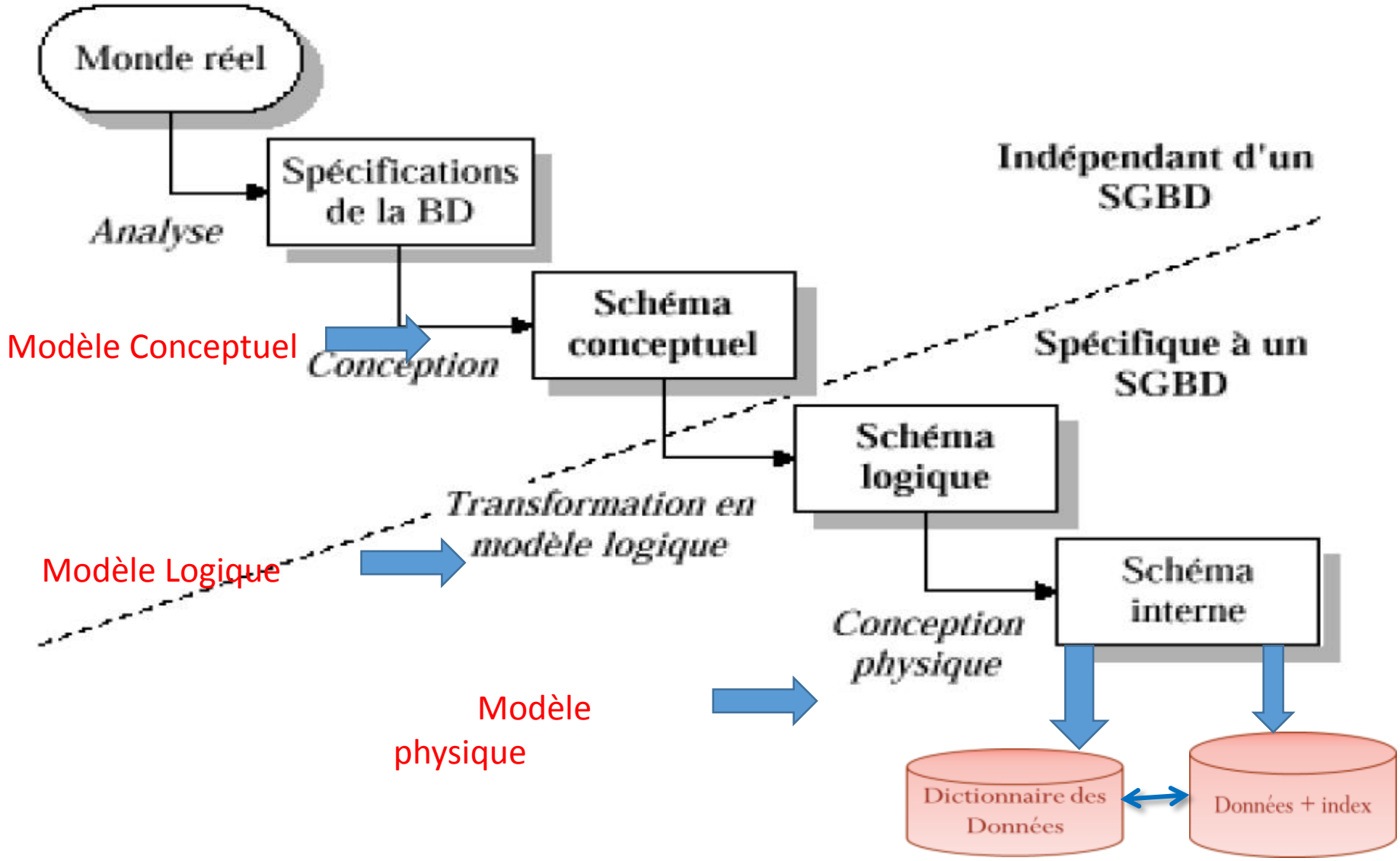
Système de Gestion de BD



- Un SGBD est un intermédiaire entre les utilisateurs et les fichiers



Cycle De Vie D'une BD (rappel)



Catalogue ou dictionnaire de données du SGBD

- Présent dans tout SGBD multi-bases et multi-utilisateurs : essentiel à l'administration de la base
- Base de données spéciale, propriété du SGBD, gérant tous les objets (BDs, tables et leurs contraintes, vues, utilisateurs, droits, index, etc. …) connus du SGBD
- On parle aussi de tables Système, de dictionnaire de données, de méta-base
- Structure
  - Une ou plusieurs tables (ou vues) par type d'objet géré par le SGBD
- Pour chaque relation
  - nom de la relation, identificateur du fichier et structure du fichier
  - nom et domaine de chaque attribut
  - nom des index
  - contraintes d'intégrité (clé primaire, clés étrangères, …)
- Pour chaque index
  - nom et structure de l'index
  - attribut appartenant à la clé de recherche
- Pour chaque vue
  - nom de la vue
  - définition de la vue
- Les utilisateurs et les autorisations d'accès
- ATTENTION : TOUS LES SGBDR NE PROPOSENT PAS LE MÊME CATALOGUE

Le catalogue (suite)

- On trouve également des données statistiques utilisées dans l'optimisation des requêtes SQL:
  - Cardinalité de chaque relation
  - Nombre de pages de chaque relation
  - Nombre de valeurs distinctes de clé de recherche pour chaque index
  - Hauteur des index de structures arborescentes (B_tree)
  - Valeurs min et max de chaque clé de recherche dans chaque index
- Exercice libre
  - Etudier et manipuler le catalogue d'un SGBDR de votre choix: Oracle, MySQL, PostgreSql
- ATTENTION : tous les SGBDR ne proposent pas le même catalogue

Exemple: le catalogue d'Oracle

- Le catalogue Oracle est organisé en Trois types de vues interrogeables
- Vues préfixées par « dba_ » Listent les informations sur tous les objets de la BD.
  - Seuls les administrateurs (sys, system, . . .) peuvent interroger ces vues.
  - Ex. : dba_tables liste toutes les tables de la BD.
- Vues préfixées par « all_ »
  - Listent les informations sur les objets accessibles par l'utilisateur courant.
  - Ex. : all_tables liste les tables que l'utilisateur peut manipuler.
- Vues préfixées par « user_ »
  - Listent les informations sur les objets possédés par l'utilisateur courant.
  - Ex. : user_tables liste toutes les tables possédés par l'utilisateur courant.

Exemple du catalogue Oracle

- La vue dict liste toutes les vues du méta-schéma, dont :

Table ou vue	Description
all_catalog	Liste des objets accessibles par l'utilisateur courant
all_users	Liste des utilisateurs créés sur l'instance courante
user_segments	Informations sur l'espace disque occupé par les objets
user_ts_quotas	Quotas fixés sur les tablespaces de l'utilisateur courant
user_objects	Objets créés par l'utilisateur courant
user_tables	Tables créées par l'utilisateur courant
user_tab_columns	Colonnes des tables créées par l'utilisateur courant
user_constraints	Contraintes créées sur des tables de l'utilisateur courant
user_cons_columns	Colonnes ciblées par les contraintes créées sur des tables
user_procedures	Procédures et fonctions créées par l'utilisateur courant
user_triggers	Déclencheurs créés par l'utilisateur courant
user_views	Vues créées par l'utilisateur courant

Table ou vue	Description
user_sys_privs	Privilèges « système » octroyés à l'utilisateur courant
user_tab_privs	Privilèges « objet » octroyés/reçus à/par l'utilisateur courant
user_tab_privs_recd	Privilèges reçus par le compte courant par d'autres
user_tab_privs_made	Privilèges octroyés par le compte courant à d'autres
user_role_privs	Rôles octroyés à l'utilisateur courant

Notion de modèle de données (rappel)

- 3 concepts permettant de décrire une BD (ce qu'on désigne par modèle de données)
  - Des structures de données pour organiser les données qui vont figurer dans la future BD
  - Des contraintes d'intégrité que doivent satisfaire ces données
    - Contraintes structurelles (imposées par le modèle)
    - Contraintes applicatives (relatives à l'application elle-même)
  - Des opérateurs permettant de manipuler ces données (ca dépend de la nature du modèle)
- Exemple du modèle relationnel
  - Structures de données: Une seule structure: la table
  - Les contraintes d'intégrité structurelles
    - Contrainte d'unicité, Contrainte d'entité, Contraintes référentielles
  - Les opérateurs: L'algèbre relationnelle traduite sous forme de commandes SQL → Sélection, projection, union, produit cartésien, jointures, …

Classification des modèles

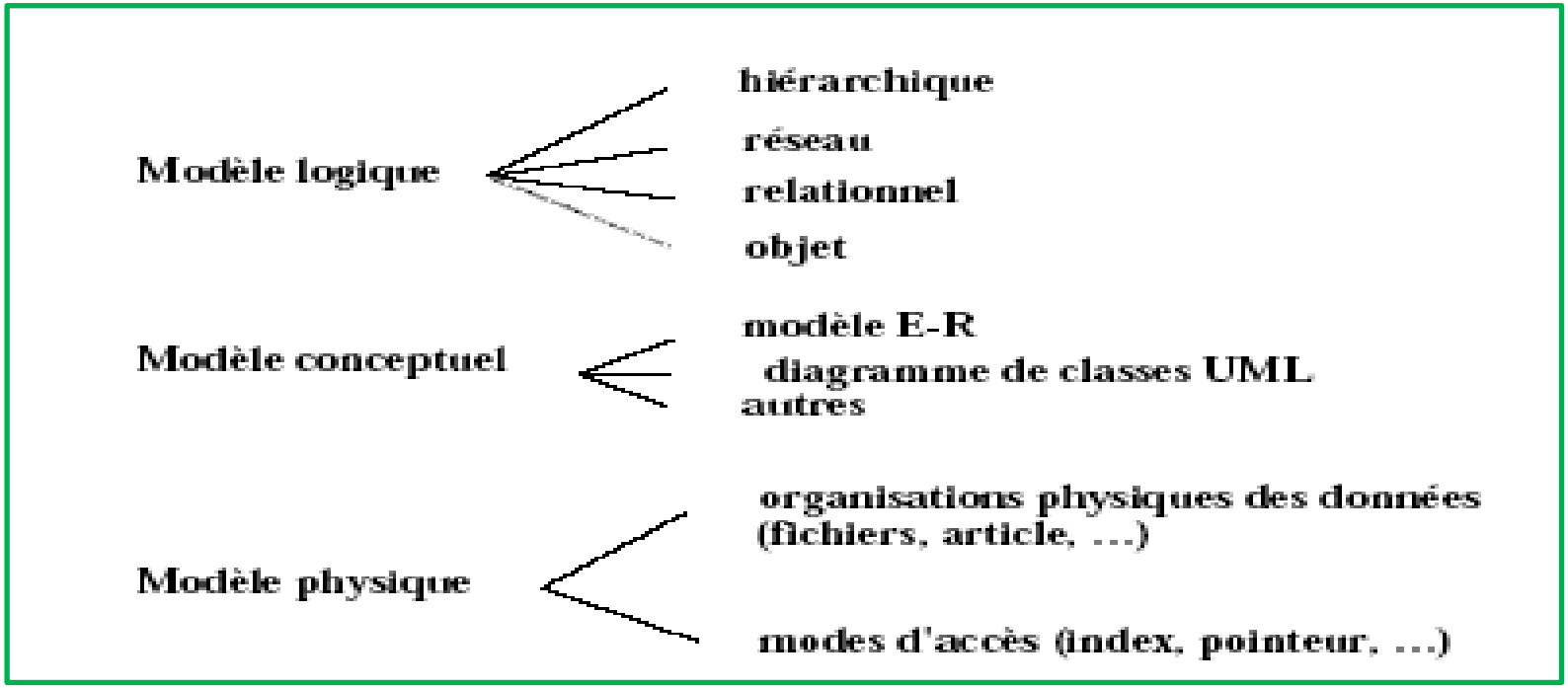
Modèles conceptuels

- description de la bd à un niveau abstrait proche de la perception des utilisateurs → **usage limité à la conception de la BD**

Modèles d'implémentation ou modèle logique

- description de la BD au niveau application Par exemple, des tables dans le modèle relationnel.

Modèle physique: description de la BD au niveau interne (fichiers)



Historique des modèles d'implémentation

- Historiquement, les modèles d'implémentation ont été définis selon l'ordre chronologique suivant:
  - Modèle hiérarchique (structure de données arbre)
  - Modèle réseau (structure de données: graphe)
  - Modèle relationnel (structure de données: tableau de lignes ou nuplets)
  - Modèle objet (structure de données: classes, attributs, méthodes)

Types de SGBD Par modèle de données

- 1ère génération 1950 – 65: Approche par les fichiers
  - Ensemble de fichiers séparés utilisés par des systèmes de gestion de fichiers (SGF) plus ou moins sophistiqués.
- 2ème génération 1965 – 70: SGBD navigationnels
    - Hiérarchique (IMS), Réseau (Codasyl), Pseudo-relationnel
    - on ne pouvait pas interroger une BD sans connaitre le schéma physique de la BD (on "navigue dans des listes sur disque")
- 3ème génération depuis 1969 : SGBD relationnel
  - (DB2, Oracle, Informix, MsAcess…)
  - Les SGBD relationnels dominent le marché.
- 4ème génération
  - SGBD OO 1990 - 1999
    - En pratique : une impasse (O2, Objectstore, Objectivity..)
  - SGBD relationnel – objet (RO) depuis 1993 - …
    - Évolution probable de tout SGBD relationnel (Oracle9i et +)
- 5ème génération: BD noSQL, Big Data, Cloud Computing ??

Exemple de SGBD selon le modèle

Les standards et quelques produits

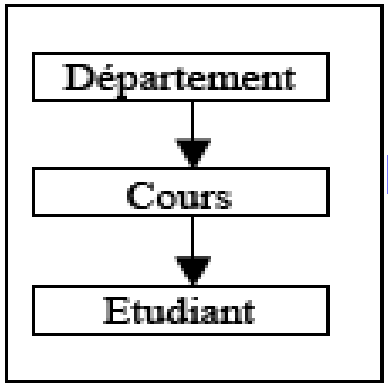
Modèles	Produits	Normes	Apparition
Hiérarchique	IMS/DL1		Années 1960
	System 2000		
Réseau	IDMS	CODASYL ANSI/SPARC	Années 1970
	IDS		
	Socrate		
	Total		
	MDBS		
Relationnel	Oracle	SQL	Années 1980
	DB2		
	Ingres		
	Informix		
	Sybase		
	SQL Server		
	mySQL		
Objet	Versant	ODMG	Années 1990
	GemStone		
	ObjectStore		

Relationnel-objet

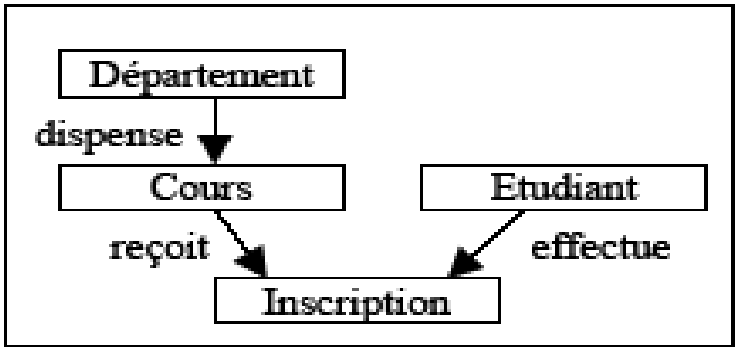
35

Akoka-Wattiau

Exemple de représentation selon le modèle



hiérarchique



réseau

La scolarité

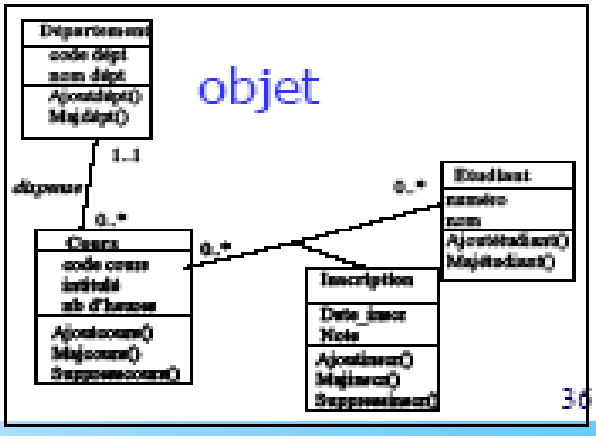
Département	code dépt	nom dépt
	M	Math
	I	Informatique

Cours	code cours	intitulé	nb d'heures	code dépt
	A12	algèbre	100	M
	G11	géométrie	50	M
	A15	analyse	120	M
	I50	programmation	100	I

Etudiant	numéro	nom
	14	Jean
	18	Jacques
	25	Pierre
	27	Paul

relationnel

Inscription	code cours	numéro	date inscription	note
	A12	14	12/12/2002	12
	A12	18	10/10/2002	10
	G11	27	10/10/2002	18
	G11	25	10/11/2002	9
	G11	14	08/09/2002	18



objet

Akoka-Wattiau

Principe d'indépendance physique

- La manipulation des données doit se faire indépendamment de leur structure de stockage
  - Représentation abstraite des données
  - Structure logique ou conceptuelle != structure physique
- On peut modifier la structure physique sans que les utilisateurs soient affectés et sans modification des programmes

Exemple :

Structure logique

Employé : Nom
Adresse
Fonction
Enfants < Prénom, Age

Structure physique

- 1 seul fichier

ou bien

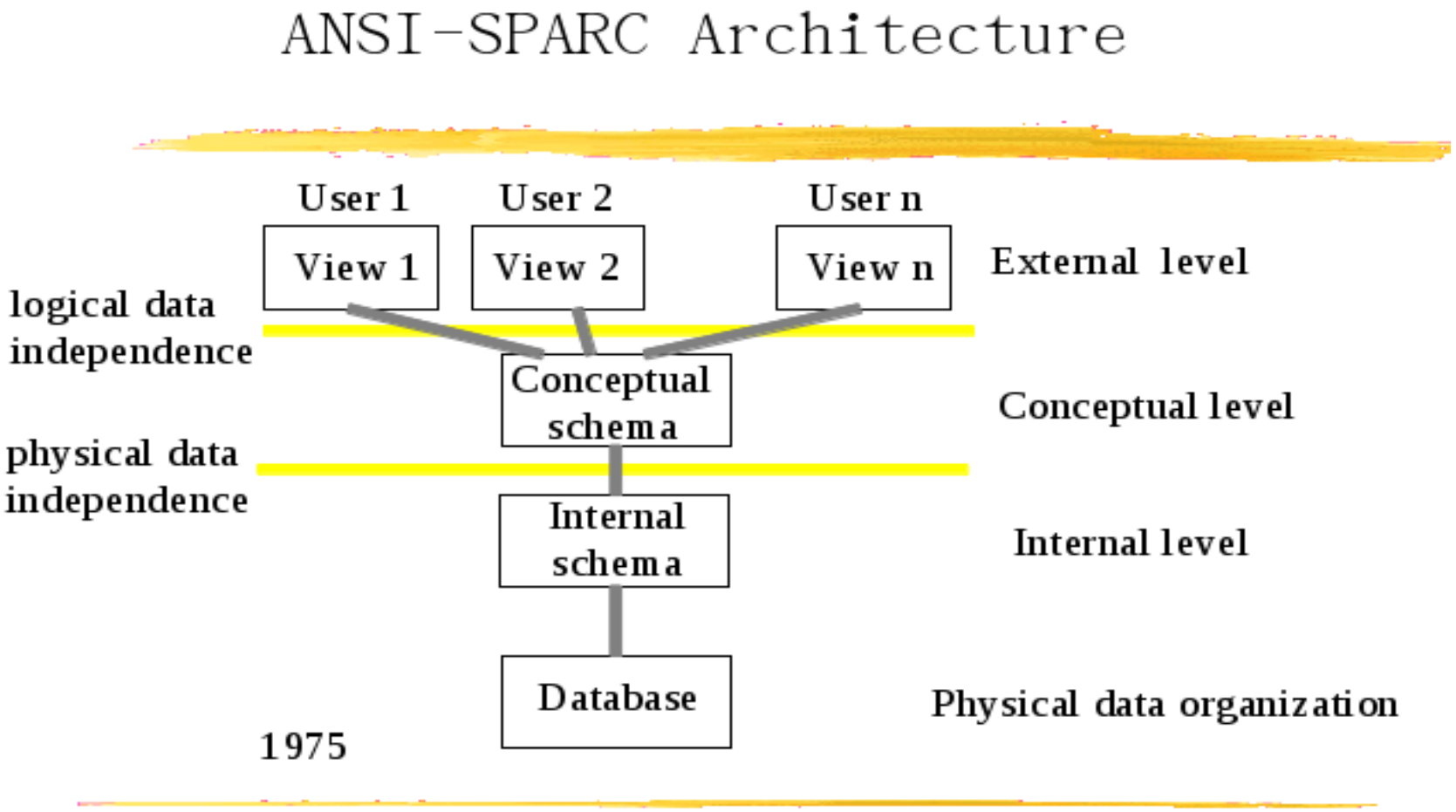
- 1 fichier Employé et 1 fichier Enfants et des pointeurs entre les deux

Principe d'indépendance logique

- Une application travaille sur une représentation logique des seules données qu'elle manipule
- Le SGBD assure la correspondance avec les données réelles (mapping logique → physique)
- Les programmes ne doivent pas être revus lors de la modification de données qu'ils n'utilisent pas.

- **Exemple**: Les données d'un hôpital: médecins, malades, chambres, etc.
  - Application n° 1 : Suivi des malades (Nom, N° SS, N° Chambre, Médecin)
  - Application n° 2 : Données d'un médecin (Nom, N°SS, N°Chambre, Thérapie)
  - Application n° 3 : Gestion du personnel : les médecins (Nom, Grade, Spécialité, Salaire)

Architecture ANSI/SPARC



Architecture ANSI/SPARC

- Niveau interne dépendant du SGBD
  - Organisation physique des données

- Niveau conceptuel indépendant du SGBD
  - Vue abstraite des données
  - On y distingue le niveau logique, dépendant du type de SGBD (hiérarchique, réseau, relationnel, objet) du niveau conceptuel
  - plus proche de l'utilisateur

- Niveau externe
  - Vue partielle des données, sous-ensemble du niveau conceptuel, pour une application ou un groupe d'utilisateurs

Architecture ANSI/SPARC

- Le schéma conceptuel (parfois appelé le schéma logique)
  - décrit les données selon le modèle de données du SGBD.
- Dans un SGBD relationnel
  - le schéma conceptuel décrit toutes les tables qui sont stockées dans la BD.

- Exemple d'une BD d'une université
  - Les tables contiennent des informations
    - sur les entités, comme les étudiants et les professeurs, etc.
    - sur les associations, telles que l'inscription des élèves dans les cours.
    - Tout étudiant est décrit par un enregistrement (un nuplet) dans la table ETUDIANTS.

Architecture ANSI/SPARC

- **Le schéma physique**
  - ajoute des détails de stockage supplémentaires
  - décrit la façon dont les tables décrites dans le schéma conceptuel sont effectivement stockées sur le disque
  - Les organisations à utiliser pour stocker les fichiers correspondants aux tables
  - Les index pour accélérer les opérations de recherche des données.

- Exemple de schéma physique
  - Stockez toutes les tables dans des fichiers non triés.
  - Créer des index sur les colonnes étudiants, professeurs et les associations avec les Cours,
  - Créer des index sur les colonnes sal de la Faculté, et capacité des salles.

Architecture ANSI/SPARC: Les schémas externes

- **Les Schémas externes**
  - *correspondent* (à peu près) aux vues dans le modèle relationnel
- Elles permettent l'accès à des données personnalisées accédées par des utilisateurs individuels ou groupes d'utilisateurs (service scolarité, service personnel, service financier, etc.).
- Chaque schéma externe se compose d'une collection de tables et de vues du schéma conceptuel.
- Remarque: Toute BD a exactement
  1. Un schéma conceptuel
  2. Un seul schéma physique car il n'a qu'un seul ensemble de relations stockées
  3. Plusieurs schémas externes, chacun adapté à un groupe particulier d'utilisateurs.

Architecture interne d'un SGBD

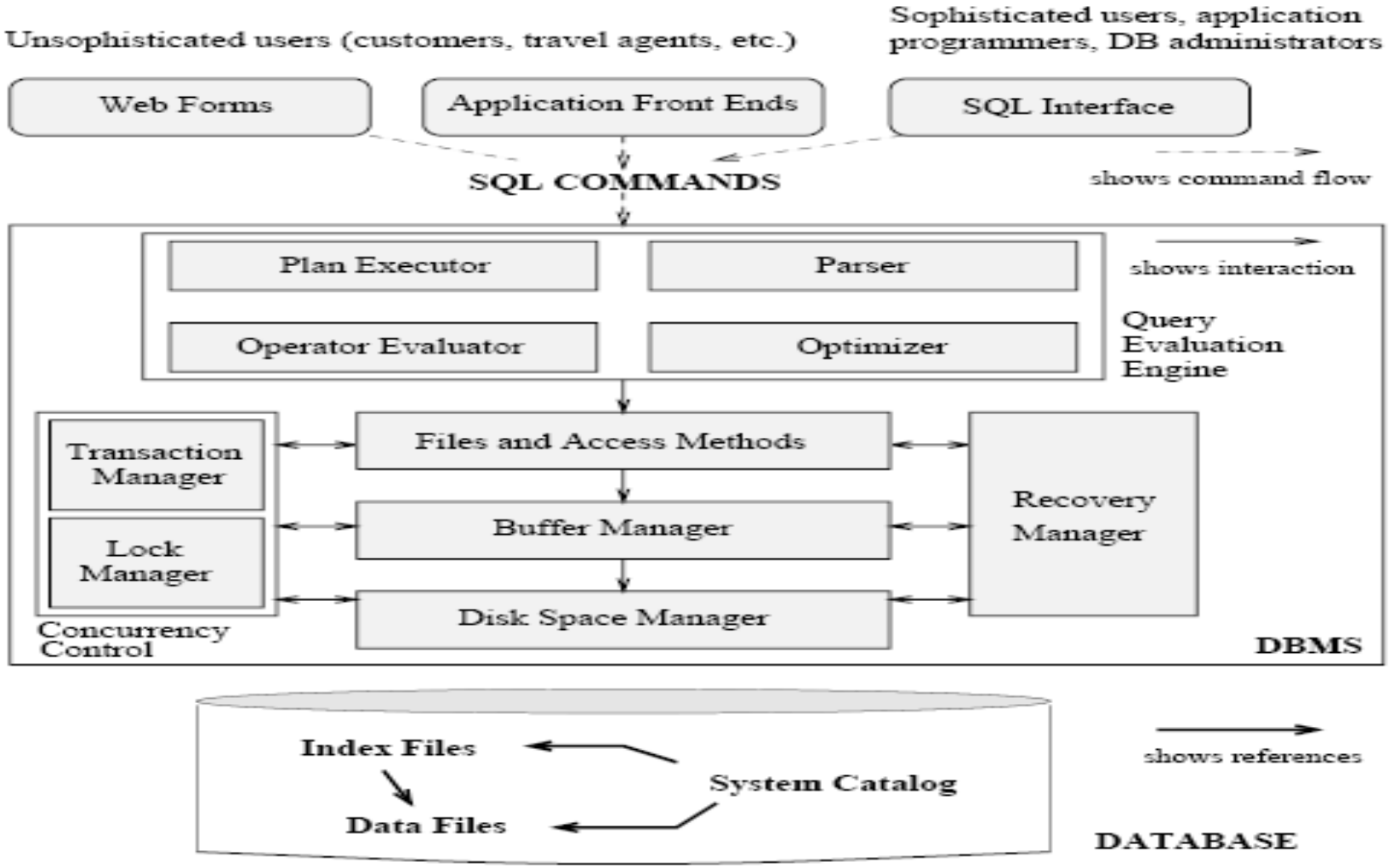


Figure 1.3 Architecture of a DBMS

Architecture interne d'un SGBD

- Le SGBD accepte les commandes SQL
  - générées à partir d'une variété d'interfaces utilisateur
  - produit (parser) des plans d'exécution
  - exécute le + performant (le moins d'accès disque) sur la B/D
  - et retourne les réponses.

- Remarque:
  - les commandes SQL peuvent être intégrés dans un langage hôte (les programmes d'application, par exemple, des programmes Java ou C/C++) → **Embedded SQL**

- Le plan d'exécution est un modèle pour l'évaluation d'une requête,
  - Représenté comme un arbre binaire (arbre algébrique)
  - Nœuds internes: opérateurs relationnels
  - Nœuds feuilles : tables (ou vues) de la BD
  - Exemple !

Architecture interne d'un SGBD

- L'optimiseur (optimiser)
  - Quand un utilisateur émet une requête, la requête est analysée puis transmise à un optimiseur de requêtes, qui utilise des informations sur la BD pour déterminer la façon la moins couteuse et la + efficace (diminue les accès disque) pour évaluer la requête → Meilleur Plan d'exécution

- L'évaluateur d'opérateurs (Operator Evaluator)
  - Le code qui implémente les opérateurs relationnels se trouve en amont de la couche fichiers et des méthodes d'accès (jointure, sélection, projection, tri, etc.)

Architecture interne d'un SGBD

- Fichiers et méthodes d'accès (Files and Access Methods)
  - Un fichier, dans un SGBD, est une collection de pages qui contiennent elles mêmes une collection d'enregistrements.
  - Cette couche exploite typiquement des fichiers organisés selon plusieurs méthodes (ISAM, B-Arbre, Hachage, Bitmap, etc.) , ainsi que des pages des fichiers d'index.

- Cette couche garde une trace des pages de chaque fichier et organise les données dans les pages (voir + loin) selon différentes techniques.

- Remarques:
  - Les Problèmes de codage interne des données et des enregistrements dans les pages des fichiers sont examinées dans un chapitre à venir.
  - Les organisations des fichiers et des index nécessite plus qu'un chapitre.

Architecture interne d’un SGBD

- Gestionnaire du buffer BD (Buffer manager)

  - La couche fichiers et méthodes d'accès se trouve au niveau de la couche « buffer manager », qui ramène les pages à partir du disque vers la mémoire principale ou inversement, au besoin, en réponse à des demandes de lecture ou d’écriture.

- Gestionnaire de l'espace disque (disk space manager).

  - La couche la plus basse du SGBD gère l'espace sur le disque où les données sont stockées. Les couches supérieures allouent, libèrent, lisent et écrivent des pages à travers des routines fournies par le gestionnaire de l'espace disque

Architecture interne d'un SGBD

- Transaction Manager et Recovery Manager
  - Contrôle de la concurrence avec le verrouillage (en général)
  - réparation des pannes
  - Le SGBD gère la concurrence et la réparation des pannes en ordonnançant soigneusement les demandes des utilisateurs et en maintenant un journal (le log) de toutes les modifications apportées à la BD.

- Le contrôle de concurrence et la réparation des pannes comportent un gestionnaire de transactions qui garantit que les transactions acquièrent et libèrent les verrous selon un protocole bien précis de verrouillage

- Une transaction est une suite de commandes SQL
  - Vérifiant les 4 propriétés ACID (voir chapitre plus loin)
  - Ressemble vaguement à un processus dans un système d'exploitation (attention, ce n'est pas exactement la même chose)

Les propriétés ACID d'une transaction (voir chapitre plus loin)

- Atomicité
  - – Soit toutes les mises à jour sont validées soit aucune (tout ou rien)
- Cohérence
  - – La base doit passer d'un état cohérent à un autre état cohérent
- Isolation
  - – Les résultats d'une transaction ne ne sont visibles que lorsque cette transaction est validée
- Durabilité
  - – Les résultats d'une transaction validée (commit) sont définitivement sauvegardées sur disque et ne sont pas perdues en cas de panne

Architecture interne d'un SGBD

- le gestionnaire de verrous (Lock Manager)
  - garde une trace des demandes des verrous sur les objets de la BD et de leur libération.
  - Le gestionnaire des pannes est responsable du maintien du log et la restauration de la BD à un état cohérent après un crash.

- Remarques
  - Le Gestionnaire de l'espace disque et le gestionnaire du buffer BD et des fichiers doivent interagir avec ces composants.
  - Nous discutons du contrôle de la concurrence et de la réparation des pannes détail dans des chapitres à venir.

Les Architectures possibles des BD

- Les technologies des dernières années ont amené la notion d'environnement distribué (dispersions des données).
- Pour relier plusieurs ordinateurs entre eux, nous utilisons :
  - Réseaux téléphoniques
  - Réseaux grand débit
  - Liaisons satellite
- Ce changement a favorisé une nouvelle approche, basée sur la technologie n-tiers
- Le partage d'informations a modifié les politiques des entreprises en matière de répartition et de partage des données

BASES DE DONNÉES LOCALES

- Certaines bases de données sont constituées afin de satisfaire un seul utilisateur.
- Ces bases de données sont exploitées généralement sur microordinateurs.
- Une excellente ergonomie
  - Ils utilisent au mieux les capacités des interfaces graphiques modernes.
- Elles sont puissantes
  - mais leurs capacités sont souvent freinées par les possibilités du micro-ordinateur sous-jacent.

- Exemple
  - ACCESS de Microsoft, Visual DBASE de Borland, PARADOX de Borland repris par l'éditeur Novell, APPROACH de Lotus, 4D de ACI, FILEMAKER PRO de Claris ...

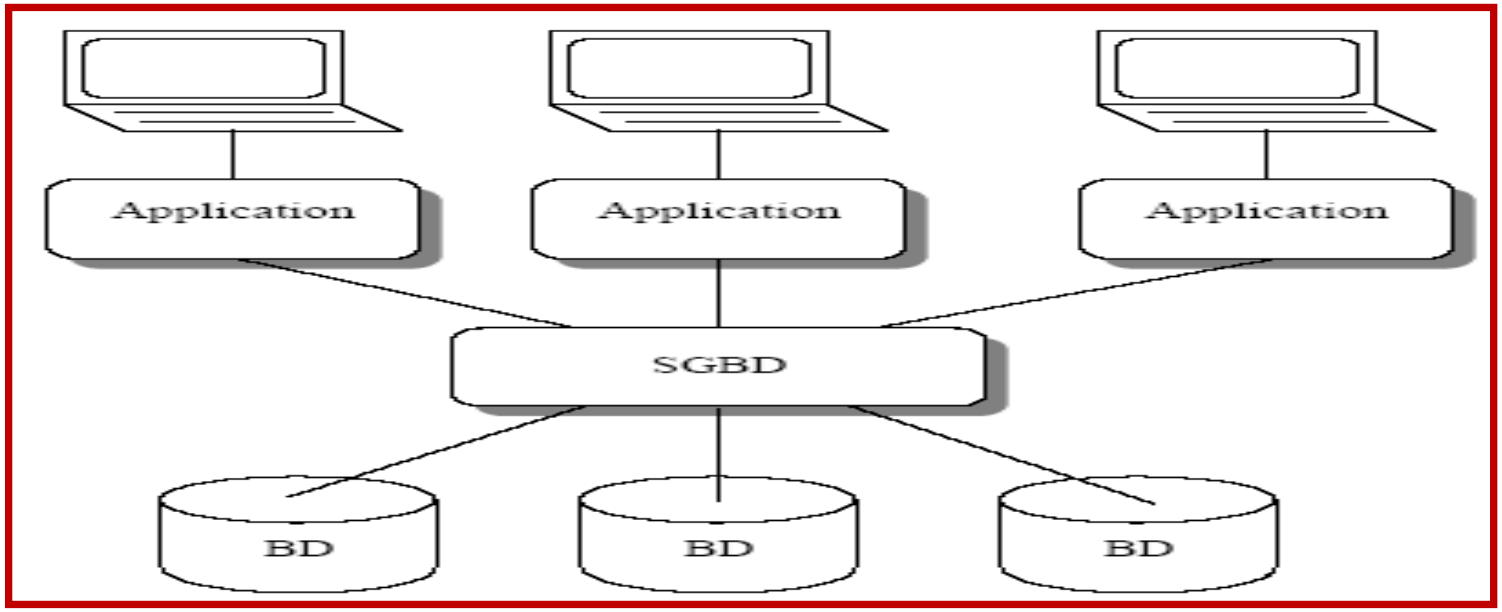
BASES DE DONNEES REPARTIES

- Selon la quantité de données à mémoriser
- Selon le nombre d'utilisateurs qui doivent être connectés simultanément.
- Sur un serveur, appartenant à la classe des mini ou grands ordinateurs.
- Cette base de données sera rendue disponible aux utilisateurs par interrogation d'un serveur
  - Via un réseau informatique ou par connexions téléphoniques.

- 3 architectures possibles
  - – architecture centralisée
  - – Plusieurs architectures client-serveur (n-tiers, etc.) ?
  - – architecture intranet

BD CENTRALISÉE

- la plus ancienne → Programmes d'application et SGBD sur même machine (même site)
- composée d'ordinateurs centraux + de terminaux
- Tout le travail (les processus) s'exécute sur les systèmes centraux, → le temps de réponse aux requêtes dépend de la charge du système.
- Ce sont des systèmes simples mais peu flexibles



BD RÉPARTIE

- Chaque site a une version complète du SGBD
- Données réparties sur les sites (Fragmentation)
  - Fragmentation horizontale : les sites ont tous les mêmes relations avec différents contenus (voir découpage)
    - unions des données sur les différents sites
  - Fragmentation verticale : les sites ont des relations différentes, tous connaissent la structure de la BD (voir découpage)
    - Jointures des données sur les différents sites
  - Fragmentation mixte
- Pas d'opérateur spécifique pour accéder aux données (SQL suffit)

FRAGMENTATION HORIZONTALE

- La fragmentation des données implique le découpage d'une relation en plusieurs sous-relations pour chaque site donné.
- Il existe deux façons de fragmenter: *horizontalement* ou *verticalement*.

Exemple de la relation Dépôt

Agence	Compte	Client	Position
Hillside	305	Lowman	500
Hillside	226	Camp	336
Valleyview	117	Camp	205
Valleyview	402	Kahn	10000
Hillside	155	Kahn	2
Valleyview	408	Kahn	1123
Valleyview	639	Camp	750

FRAGMENTATION HORIZONTALE

Par exemple, on peut subdiviser la relation Dépôt par agence:

$Dépôt_1 = \sigma_{agence = Hillside}$ (Dépôt)

Agence	Compte	Client	Position
Hillside	305	Lowman	500
Hillside	226	Camp	336
Hillside	155	Kahn	62

La relation $Dépot_1$ est stockée à l'agence Hillside

$Dépôt_2 = \sigma_{agence = Valleyview}$ (Dépôt) :

Agence	Compte	Client	Position
Valleyview	117	Camp	205
Valleyview	402	Kahn	10000
Valleyview	408	Kahn	1123
Valleyview	639	Camp	750

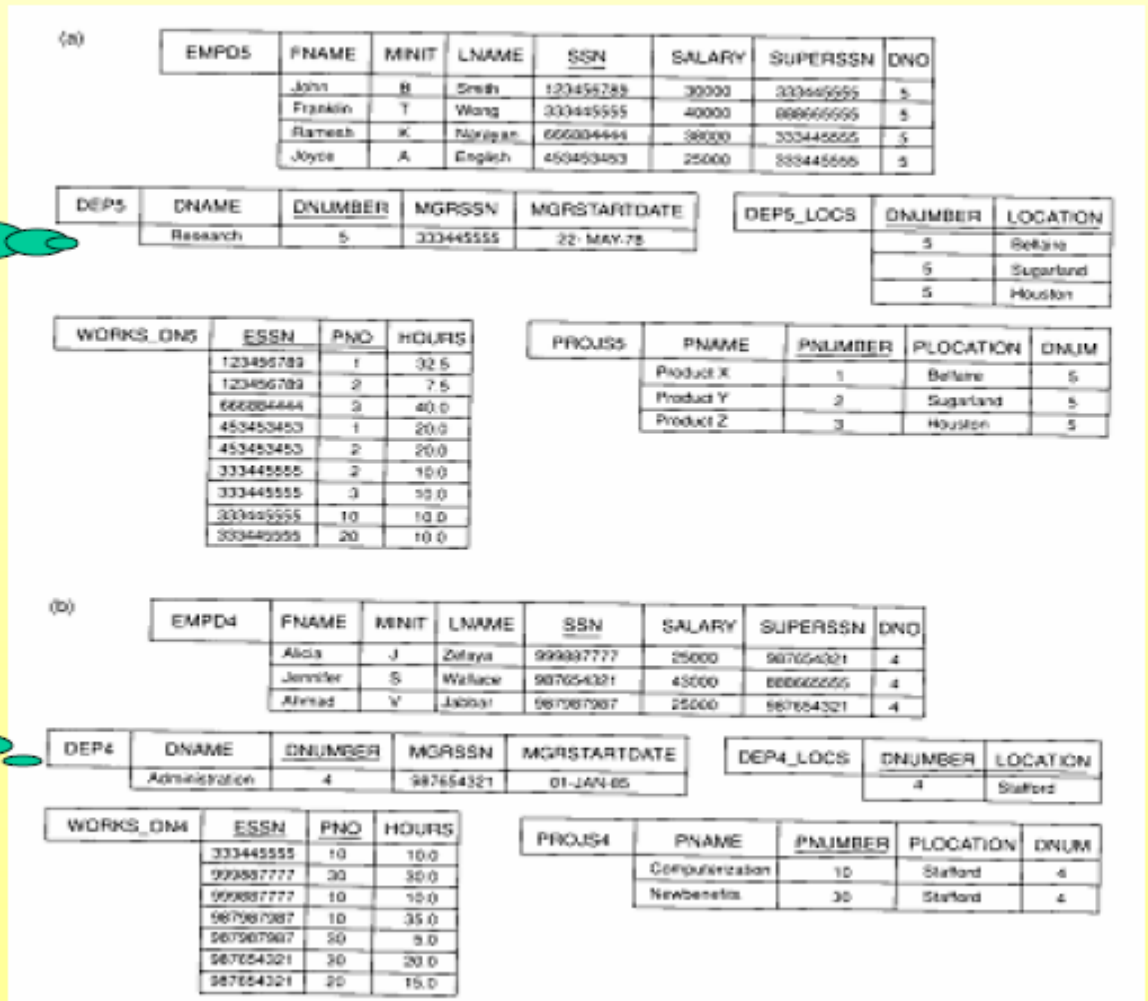
La relation $Dépot_2$ est stockée à l'agence Valleyview

FRAGMENTATION HORIZONTALE

Voici un autre exemple de fragmentation horizontale su
sites différents.

Département de recherche

Département d'administration



(a)

EMPD5

FNAME	MINIT	LNAME	SSN	SALARY	SUPERSSN	DNO
John	B	Smith	123456789	30000	333445555	5
Franklin	T	Wong	333445555	40000	888665555	5
Ramesh	K	Narayan	666884444	38000	333445555	5
Joyce	A	English	453453453	25000	333445555	5

DEP5

DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
Research	5	333445555	22-MAY-78

DEP5_LOCS

DNUMBER	LOCATION
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON5

ESSN	PNO	HOURS
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0

PROJ5

PNAME	PNUMBER	PLOCATION	DNUM
Product X	1	Bellaire	5
Product Y	2	Sugarland	5
Product Z	3	Houston	5

(b)

EMPD4

FNAME	MINIT	LNAME	SSN	SALARY	SUPERSSN	DNO
Alicia	J	Zelaya	999887777	25000	987654321	4
Jennifer	S	Wallace	987654321	43000	888665555	4
Ahmad	V	Jabbar	987987987	25000	987654321	4

DEP4

DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
Administration	4	987654321	01-JAN-85

DEP4_LOCS

DNUMBER	LOCATION
4	Stafford

WORKS_ON4

ESSN	PNO	HOURS
333445555	10	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0

PROJ4

PNAME	PNUMBER	PLOCATION	DNUM
Computerization	10	Stafford	4
Newbenefits	30	Stafford	4

FRAGMENTATION VERTICALE

- La fragmentation verticale permet de séparer une relation au niveau des colonnes (attributs).
  - Cette séparation ajoute un attribut spécial à la relation (*tuple-id)*.
- Ce *tuple-id* représente l'adresse physique du tuple (peut être une clé primaire).

Voici la fragmentation verticale de la relation Dépôt:

$Dépôt_3 = \sigma_{agence,client,tuple\text{-}id}$ (Dépôt)

Agence	Client	*tuple-id*
Hillside	Lowman	1
Hillside	Camp	2
Valleyview	Camp	3
Valleyview	Kahn	4
Hillside	Kahn	5
Valleyview	Kahn	6
Valleyview	Camp	7

Les colonnes **agence** et **client** sont conservées.

$Dépôt_4 = \sigma_{compte,position,tuple\text{-}id}$ (Dépôt)

Compte	Position	*tuple-id*
305	500	1
226	336	2
117	205	3
402	10000	4
155	62	5
408	1123	6
639	750	7

Les colonnes **compte** et **position** sont conservées.

L'attribut **tuple-id** est en général **invisible** à l'utilisateur et il sert essentiellement à recombiner les relations verticales.

Fragmentation verticale

- La fragmentation mixte est une combinaison des fragmentations horizontales et verticales.
- Par exemple, on peut fragmenter horizontalement la relation Dépôt3 :

$$Dépôt_{3a} = \sigma_{agence = Hillside} (Dépôt_3),$$

Agence	Client	*tuple-id*
Hillside	Lowman	1
Hillside	Camp	2
Hillside	Kahn	5

$$Dépôt_{3b} = \sigma_{agence = Valleyview} (Dépôt_3).$$

Agence	Client	*tuple-id*
Valleyview	Camp	3
Valleyview	Kahn	4
Valleyview	Kahn	6
Valleyview	Camp	7

Réplication et fragmentation des données

- Les techniques qui précèdent peuvent être appliquées séquentiellement à une même relation :
- Imaginons un système réparti sur 10 sites (S1, S2, ..., S10); avec la fragmentation mixte de Dépôt en :
  - Dépôt3a, Dépôt3b et Dépôt4,
- Puis stockée de cette façon:
  - une copie de Dépôt3a sur les sites S1, S3 et S7,
  - une copie de Dépôt3b sur les sites S7 et S10 et
  - une copie de Dépôt4 sur les sites S2, S8 et S9.

RÉPLICATION ET FRAGMENTATION DES DONNÉES

- Une BD relationnelle peut avoir ses relations stockées dans la BD de plusieurs façons :
  - avec réplication: les divers sites stockent chacun une copie de la relation;
  - avec fragmentation: la relation est découpée en plusieurs fragments hébergés par un site donné;
  - avec réplication et fragmentation: combinaison des deux processus précédent
- La réplication implique qu'une relation est répliquée et stockée intégralement dans deux ou plusieurs sites.
- Avantages et inconvénients:
  - disponibilité des données: une relation peut être atteinte sur n'importe quel site en cas d'avarie affectant un site donné;
  - parallélisme des traitements: pour des transactions de lectures, plusieurs sites peuvent travailler en parallèle sur une même relation;
  - servitudes de mise à jour: toutes mises à jour (écritures) sur une relation doivent être appliquées à l'ensemble des sites, ce qui alourdit la procédure de gestion de la BD

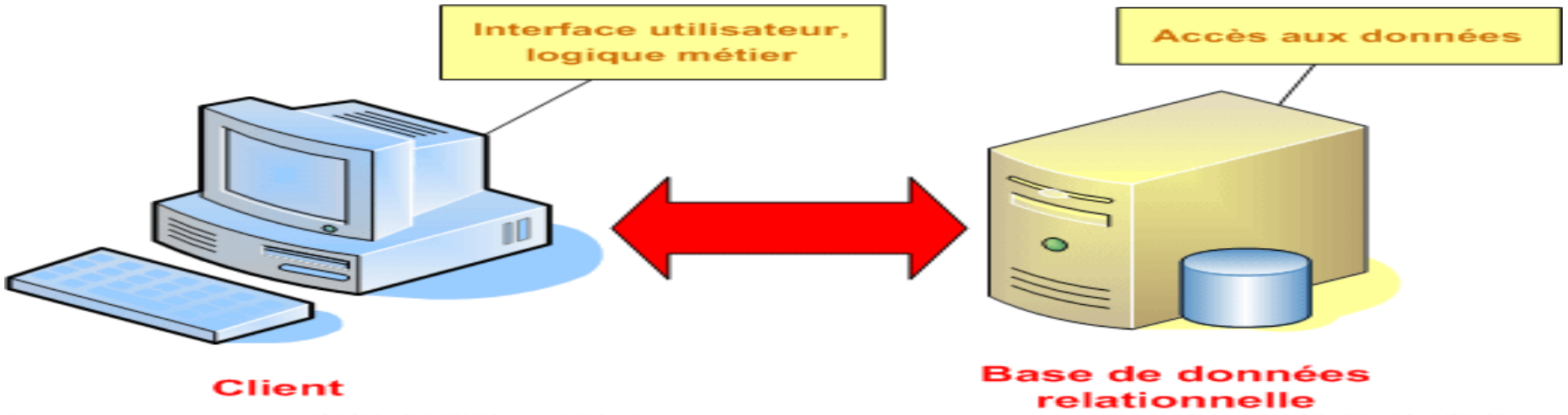
TRANSPARENCE ET AUTONOMIE

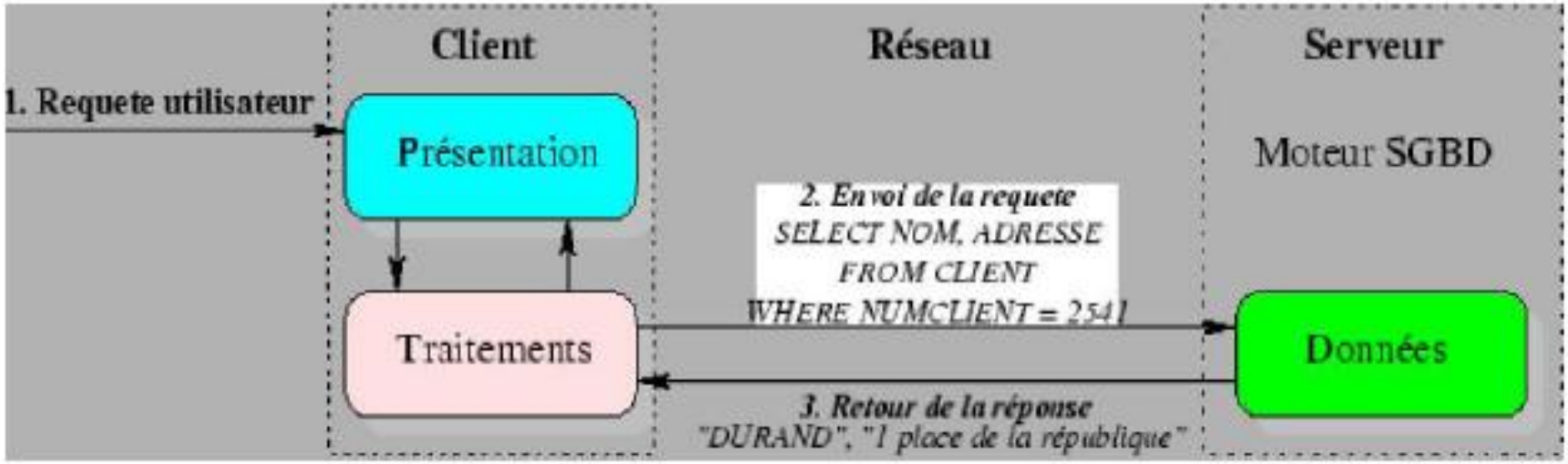
- La **dispersion** d'une relation sur plusieurs sites devrait être transparente à l'utilisateur.
- Pour obtenir l'autonomie locale, le système utilise une désignation particulière pour identifier les sous-relations réparties sur la BD.
- ***Par exemple :***
- site10.Dépôt.f3.r2 désigne, sur le site 10, la réplique 2 du fragment 3 de la relation Dépôt.
- Inconvénient: Restreint la transparence des données puisqu'elles sont marquées par leur origine.
- Solution: Création pour chaque sous-relations des alias à l'usage de l'utilisateur.
- *Ce dernier peut alors manipuler ces sous-relations au moyen d'étiquettes de substitution plus simples que le système transcrit en désignations complètes.*

LES ARCHITECTURES N-TIERS

- Web
  - Très répandu et permet de traiter différents types de données (textes, sons, images, …)
- SGBD
  - Accès efficace et sécurisé à des données
  - structurées (tables/nuplets)
- La combinaison des deux est presque systématique actuellement
- Web : manipule des données statiques en lecture seule via HTML
- Les accès à une BD supposent que :
  - les données à stocker dans les pages ne sont pas connues à priori
  - des requêtes de mise à jour sont indispensables
- Solution : générer les pages HTML à la volée (dynamique)

Architecture Client-Serveur





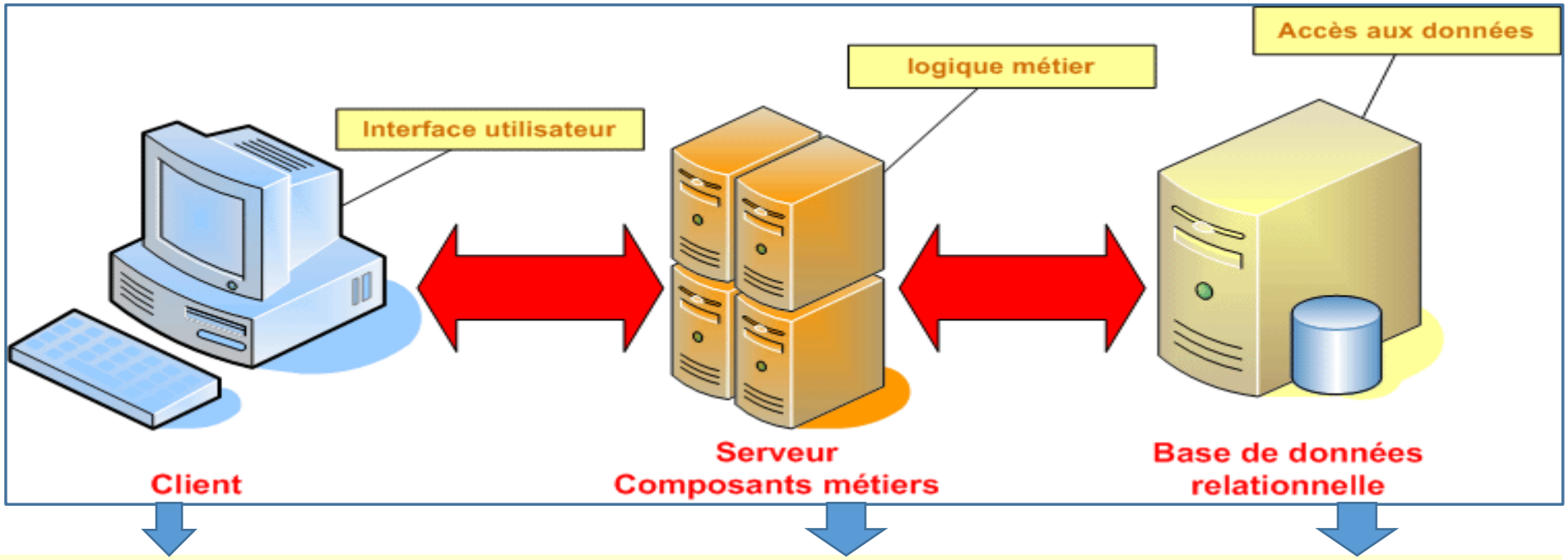
Ou architecture 2-tiers

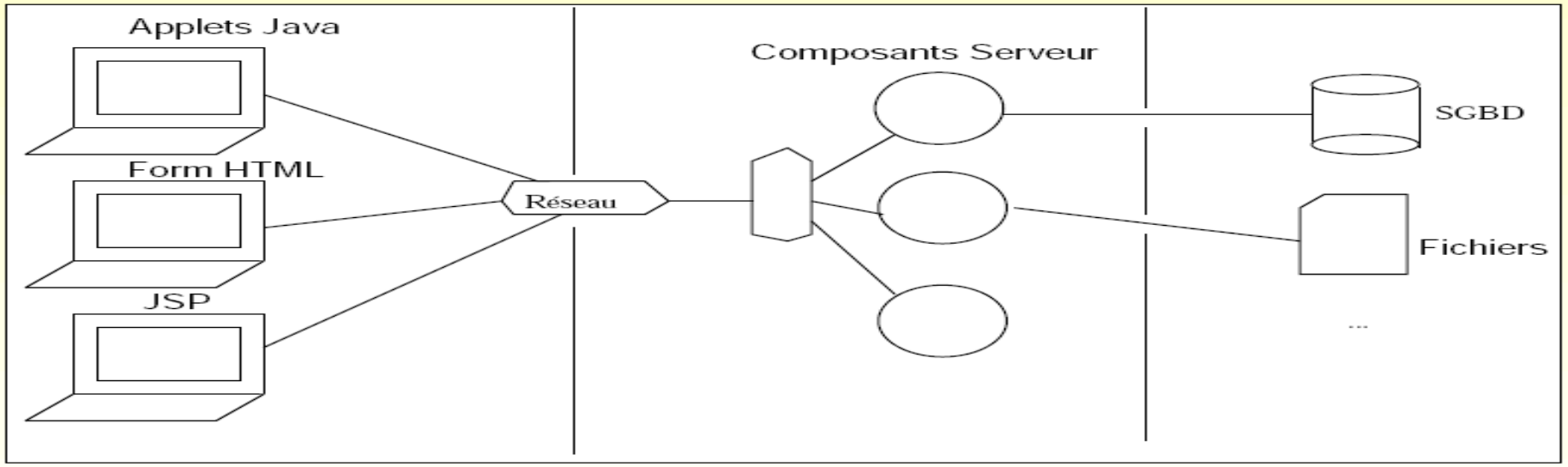
Architecture Client-Serveur

- Deux types de serveurs avec une liaison réseau éventuelle:
- Le poste client ou Serveur de transaction
  - envoie des transactions traitées par le serveur, puis traite les résultats
  - contient les outils de l'interface externe

- Le Serveur de données
  - les transactions sont exécutées par le client, le serveur ne fait que "fournir" les données
- Cet échange de messages transite à travers le réseau reliant les deux machines.
- Le client demande un service au serveur, comme par exemple la page contact.html,
- le serveur reçoit cette requête http , effectue un traitement, et renvoie la ressource demandée par le client.
- Le cas typique de cette architecture est une application de gestion fonctionnant sous Windows ou Linux et exploitant un SGBD centralisé.

Architecture trois-tiers





Architecture trois-tiers

- Le troisième niveau : les données
  - Stockage des données (SGBD, fichiers,. . .)
  - Réutilisation de code existant (ex : processus COBOL)

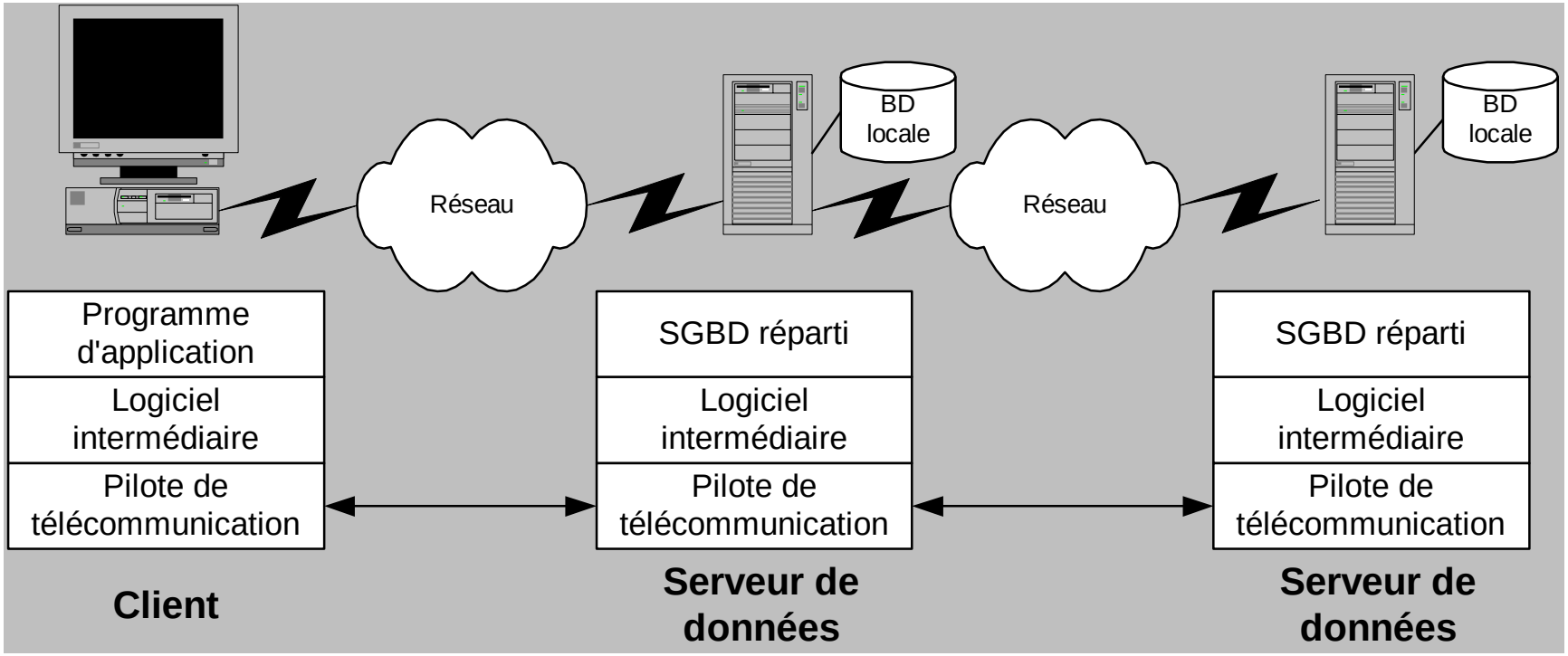
- Le second niveau : le traitement des données
  - Le programmeur gère le code métier
  - Le gestionnaire de composants gère le reste (persistance, transactions, sécurité, etc.).
- Le premier niveau : l'interface graphique
  - Uniquement l'aspect visuel
  - Pas de code métier !
  - Uniquement affichage et transfert d'informations (formulaires)
  - Plusieurs interfaces possibles pour une même application (Wap, PC, PDA,. . .)

- Un protocole privilégié : le WEB (http)
- déploiement automatique des applications !

Base de données distribuées

- Une BD distribuée se compose d'un ensemble de sites dont chacun héberge un SGBD local
- Chaque site est donc capable
  - *de traiter des* transactions locales *qui ne concernent que les données de ce site particulier.*
  - *d'exécuter des* transactions globales *sur les données de plusieurs sites.*
- Ce qui nécessite une liaison entre les sites.
- Les BD réparties communiquent au moyen de:
  - *réseaux téléphoniques,*
  - *réseau à grand débit,*
  - *liaisons par satellite.*

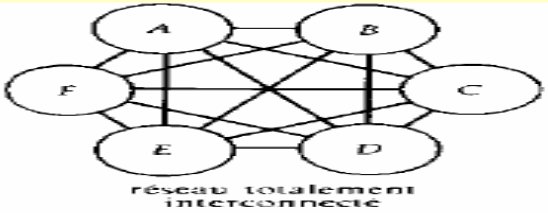
Base de données distribuées



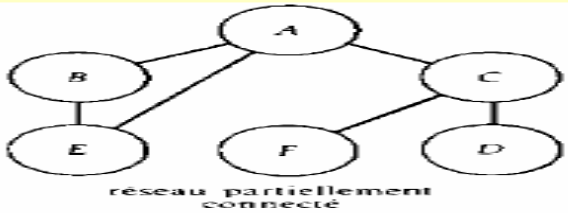
Base de données distribuées

Les configurations, les plus communément adoptées, se différencient par

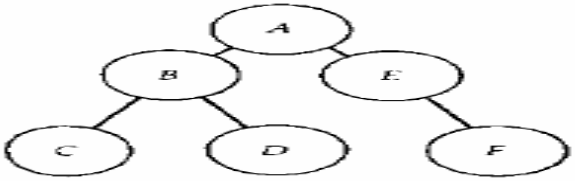
1. ***coût d'installation*** *: frais de liaison physique entre les sites;*
2. ***coût d'exploitation*** *: frais de transmission et durée de transmission*
3. ***fiabilité*** *: fréquence de pannes de liaison ou de site des liaisons établies;*
4. ***disponibilité*** *: fonctionnement continu même en cas de pannes de quelques liaisons ou de quelques sites.*



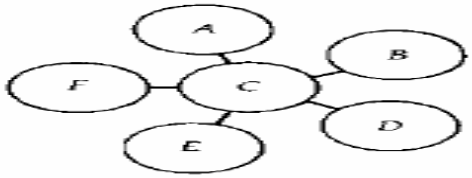
réseau totalement interconnecté



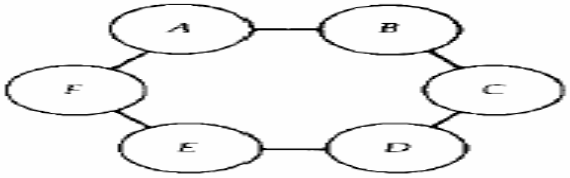
réseau partiellement connecté



réseau arborescent



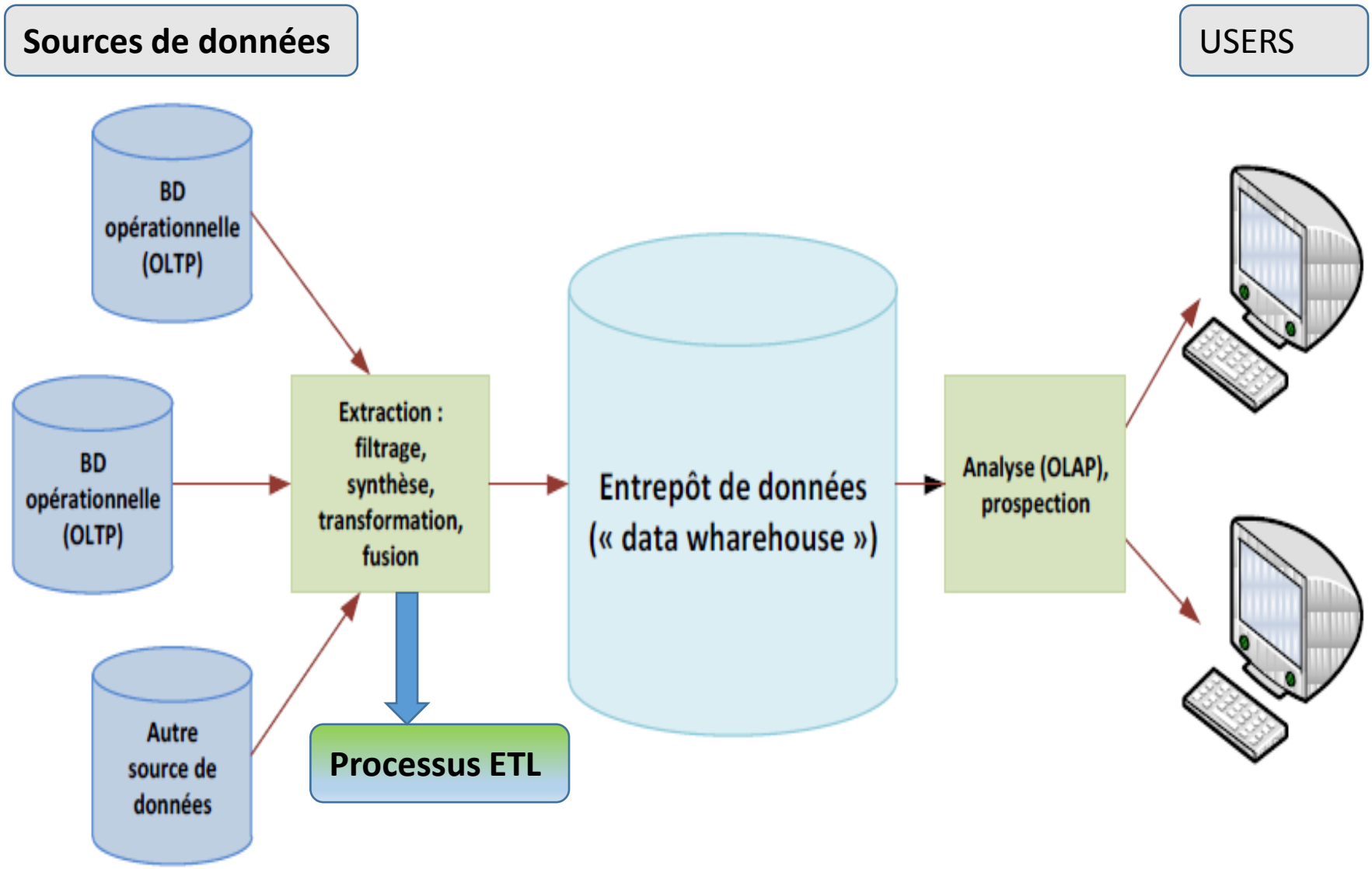
réseau étoilé



Entrepôt de données

- Entrepôt de données (BD décisionnelle, Data Warehouse)
  - BD utilisée pour collecter, ordonner, journaliser et stocker des informations provenant de plusieurs sources (BD opérationnelles, fichiers plats, fichiers Excel, etc.) et destinées aux processus d'aide à la décision en entreprise.
  - Basé, généralement, sur un SGBD relationnel
  - Proposé par William H. Inmon en 1990
  - Hébergé, en général, sur le serveur de l'entreprise, mais il l'est de plus en plus dans un Cloud.
- Traitements
  - OLTP (" On Line Transaction Processing "): traitement des données quotidiennes et récentes dans les BD opérationnelles
  - OLAP ("On Line Analytical Processing "): traitement des données collectées dans le Data Warehouse pour l'aide à la décision

Entrepôt de données



OLTP vs OLAP

Données opérationnelles	Données décisionnelles
Orientées application, détaillées, précises au moment de l'accès	Orientées activité (thème, sujet), condensées, représentent des données historiques
Mise à jour interactive possible de la part des utilisateurs	Pas de mise à jour interactive de la part des utilisateurs
Accédées de façon unitaire par une personne à la fois	Utilisées par l'ensemble des analystes, gérées par sous-ensemble
Haute disponibilité en continu	Exigence différente, haute disponibilité ponctuelle
Uniques (pas de redondance en théorie)	Peuvent être redondantes
Petite quantité de données utilisées par un traitement	Grande quantité de données utilisée par les traitements
Réalisation des opérations au jour le jour	Cycle de vie différent
Forte probabilité d'accès	Faible probabilité d'accès
Utilisées de façon répétitive	Utilisée de façon aléatoire

Tableau 1 : Différences entres les données opérationnelles et les données décisionnelles

BIBLIOGRAPHIE

- R. Ramakrishnan et J. Gehrke, Database Management Systems, Second Edition;
- McGraw-Hill, 2000, disponible à la BU 055.7 RAM
- H. Garcia Molina, J.D. Ullman et J. Widom, Database System Implementation, Prentice Hall, 2000, disponible à la BU 005.7 GAR
- H. Garcia Molina, J.D. Ullman et J. Widom, Database Systems - The Complete Book, Prentice Hall, 2002
- T. Connoly, C. Begg et A. Strachan, Database Systems A Pratical Approach to Desigh, Implementation and Management, 1998, disponible à la BU 055.7 CON
- A. Silberschatz, H.F. Korth et S. Sudarshan, Database System Concepts, McGraw- Hill, 2002, version de 1996 disponible à la BU 005.7 DAT
- C.J. Date, An Introduction aux bases de données, 6ème édition, Thomson publishing, 1998, disponible à la BU 005.7 DAT
- R.A. El Masri et S.B. Navathe, Fundamentals of Database Systems, Prentice Hall, disponible à la BU 005.7 ELM
- G. Gardarin, Bases de Données - objet/relationnel, Eyrolles, 1999, disponible à la BU 005.74 GAR + Le client - serveur, Eyrolles, 1996004.21 GAR