



IFT187 – SÉANCE 1

INTRODUCTION AUX SGBD

Nadia Tahiri, Ph. D.
Professeure adjointe
Université de Sherbrooke
Nadia.Tahiri@USherbrooke.ca

Plan de la séance

1. Introduction
2. Avantages d'un SGBD
3. Les principales architectures
4. Exemple introductif – Survol de la session
5. Références
6. Semaine prochaine



INTRODUCTION

- Pourquoi l'utilisation des bases de données ?

Persistance des données

- Dans un système informatique, les données (informations) sont manipulées par les programmes en mémoire centrale (RAM)
 - ❖ Le système d'exploitation libère l'espace mémoire alloué au programme dès qu'il termine son exécution
 - ❖ La mémoire centrale est de nature volatile : perd son contenu dès que la machine n'est plus alimentée
- Que faire si le programme se termine ?
- Lors d'une coupure de l'alimentation ?

Persistance des données

- Il est nécessaire de faire **persister** (i.e. sauvegarder) les données
- **Donnée persistante** : Une donnée dont la durée de vie dépasse celle de l'exécution du programme
- Plusieurs mécanismes de persistance, i.e. :
 - Fichiers plats
 - Fichiers structurés (i.e. : XML, CSV, JSON)
 - **Bases de données relationnelles**
 - Bases de données non-relationnelles
 - Bases de données objet

Donnée

- **Définition** : Une donnée est une représentation d'un fait à l'aide d'un code binaire stocké dans une mémoire de l'ordinateur.
- Peut être de plusieurs types : **type simple** (i.e. entier, réel, caractère), de **type complexe** (i.e. combinaison de types simples) ou multimédia (i.e. image, vidéo, son).



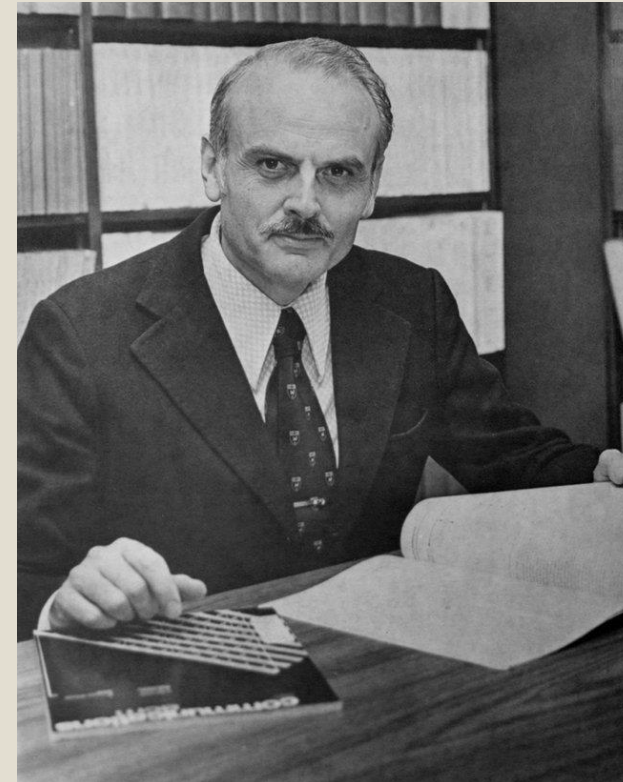
Donnez quelques exemples de données.

Base de données

- Au sens large : Une collection de données persistantes
- Au sens plus strict :
 - Ensemble de **données persistantes** qui sont
 - Fortement **structurées** (typiquement), dont la structure est définie par un **schéma**, et
 - Qui sont gérées par un **système de gestion de base de données**.

Le modèle relationnel

- Edgar F Codd
 - Inventeur du modèle relationnel
 - "A Relational Model of Data for Large Shared Data Banks", Communications of the ACM, 1970
 - Prix Turing, 1981



Base de données relationnelle

- “Une base de données où l’information est organisée dans des tableaux à deux dimensions appelés des **relations** ou **tables**, selon le modèle introduit par Edgar F. Codd.” (1970)

Source : Wikipedia

CodePerm	Nom	Prénom	DateNaiss
SNOJ1982102101	Snow	Jon	1982-10-21
STAA1987071509	Stark	Arya	1987-01-10
TARD1979071502	Targaryen	Daenerys	1979-07-15
GRET1981091501	Greyjoy	Theon	1981-09-15
LANT1970061203	Lannister	Tyrion	1970-06-12

Système de gestion de base de données (SGBD)

- **Un logiciel système**
 - servant à **stocker**, à **manipuler** ou **gérer**, et à **partager** des informations dans une base de données,
 - en **garantissant** la **qualité**, la **pérennité** et la **confidentialité** des informations,
 - tout en **cachant la complexité des opérations**.

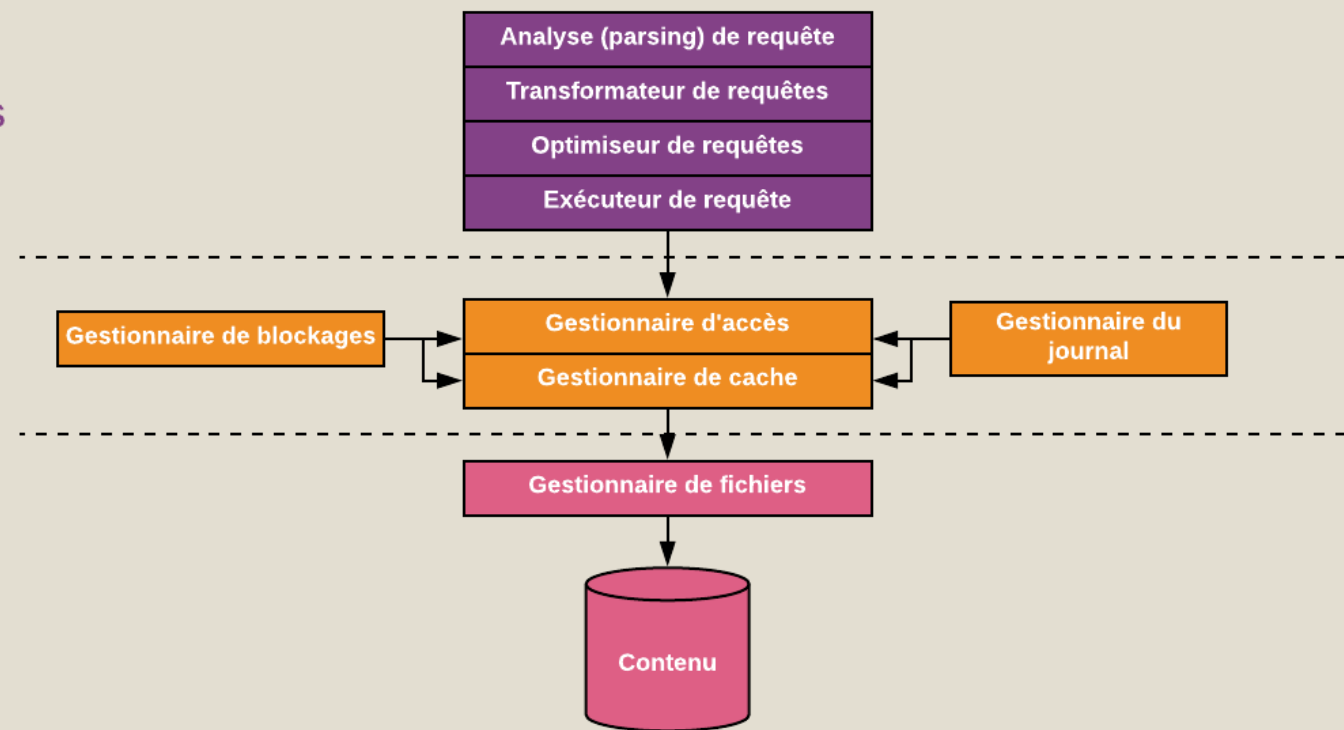
Source : Wikipedia

Architecture d'un SGBD

Traitement de requêtes

Accès aux données

Gestion du disque



Quelques SGBD bien connus

- SGBDs Relationnels :
 - Oracle Database : SGBD
 - MariaDB (ou MySQL)
 - **PostgreSQL** utilisé dans le cadre de ce cours
 - Microsoft SQL Server
- SGBDs non-relationnels (noSQL) :
 - MongoDB
 - Apache Cassandra



ORACLE®
DATABASE



MariaDB



PostgreSQL



Microsoft®
SQL Server™



mongoDB



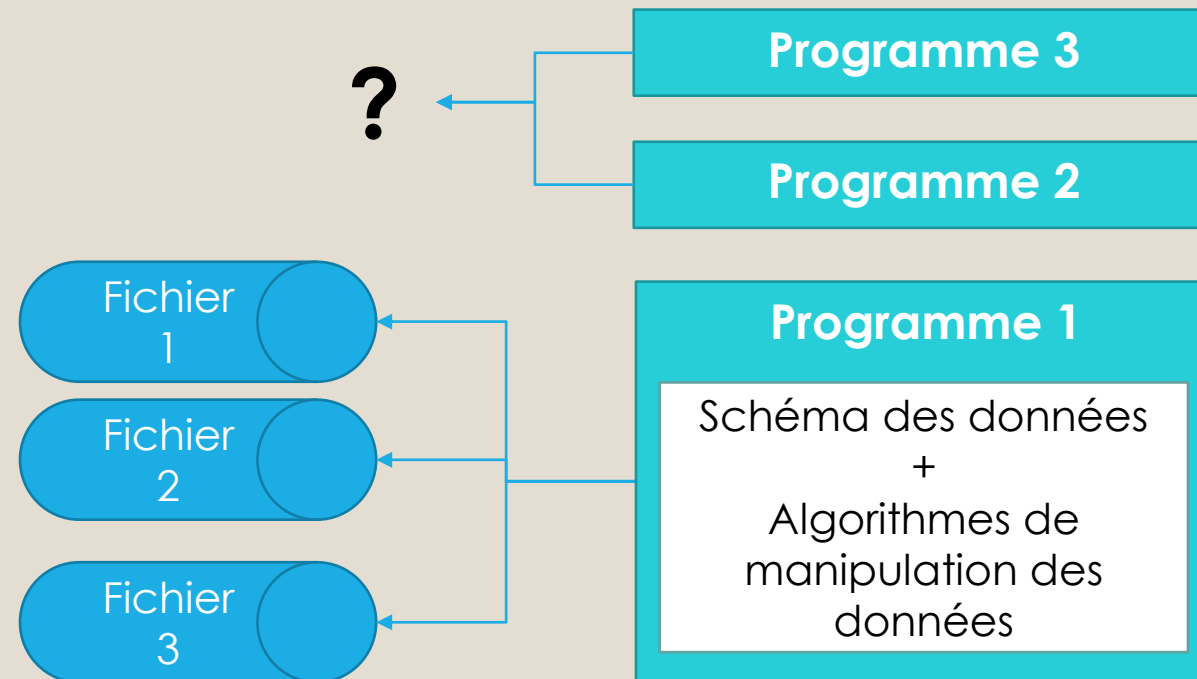
cassandra



LES AVANTAGES : SGDB VS FICHIERS

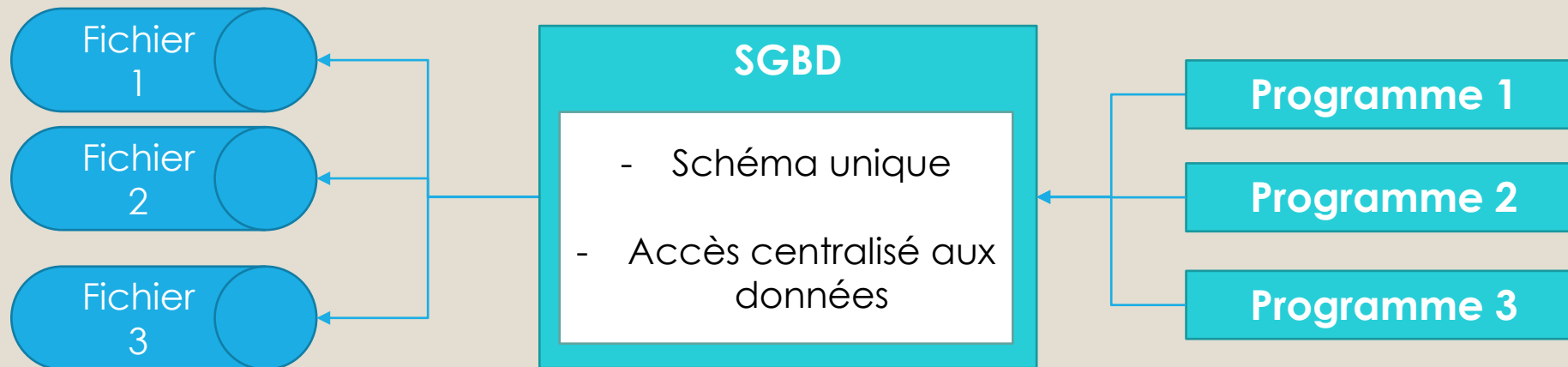
Abstraction de l'accès aux données

- **Fichiers** : Nécessité de partager un schéma *ad-hoc* et de partager (voir ré-implémenter) les algorithmes de manipulation des données



Abstraction de l'accès aux données

- **SGBD** : Schéma formel partagé, manipulation et accès aux données à travers un nœud central



Abstraction de l'accès aux données

- En utilisant un langage formel et normalisé (le SQL), le SGBD permet de :
 - Décrire la structure des données (le schéma) avec le langage de définition des données (LDD)
 - D'ajouter, modifier, supprimer et interroger les données avec le langage de manipulation des données (LMD)
 - Les programmes transmettent **les requêtes** (ordres destinés au SGBD) de définition ou de manipulation des données de façon séquentielle, le SGBD leur retourne le résultat.
 - Permet de formuler ce qu'on veut réaliser – le **QUOI** – sans se préoccuper du **COMMENT**.

Abstraction de la structure interne des données

- Le format de stockage des données (dans des fichiers) est géré par le SGBD. Les programmes en font une abstraction totale.



Contrôle d'accès aux données

- **Fichiers :**

- ❖ Nécessaire d'implémenter les divers contrôles d'accès dans le programme.
- ❖ Contrôle d'accès inter-programmes pas efficace.

- **SGBD :** Contrôle d'accès centralisé. Accès définis à l'aide du langage de contrôle de données (LCD).



Maintient de l'intégrité sémantique des données

- Lors de la définition des données, nous définissons un ensemble de **contraintes d'intégrité sémantique** des données
- **Une contrainte d'intégrité** est une règle devant **toujours** être respectée par les données de la BD.
- Exemple : Le solde d'un compte bancaire ne peut être négatif.

Maintient de L'Intégrité référentielle des données

- Les données dans une base de données sont réparties sur plusieurs tables (i.e. : une table pour les Clients, une table pour les Produits, une pour les Factures)
- Le SGBD permet de définir des liens entre les diverses tables qui constituent la base de données
- L'intégrité référentielle, garantie par le SGBD, permet de s'assurer que les liens entre les tables ne seront pas incohérents.

Maintient de l'intégrité référentielle

CodePerm	Nom	Prenom	DateNaiss
SNOJ1982102101	Snow	Jon	1982-10-21
STAA1987071509	Stark	Arya	1987-01-10
TARD1979071502	Targaryen	Daenerys	1979-07-15

- La suppression de cette ligne entrainera une incohérence référentielle.
- Le SGBD s'assure que de telles incohérences ne surviennent pas (i.e. : suppressions en cascade)

CodePerm	Epreuve	Note
SNOJ1982102101	Intra1	85
STAA1987071509	Intra1	92
TARD1979071502	Intra1	65

Contrôle des accès concurrents

- Le SGBD permet un accès multi-utilisateur aux données
- Problème d'accès concurrents et de fiabilité des données
- Le SGBD met en place des mécanismes sophistiqués pour protéger les utilisateurs des effets indésirables des manipulations de données par les autres utilisateurs.
- Ceci est réalisé grâce à un mécanisme de gestion transactionnelle

Fiabilité

- Une panne matérielle peut facilement corrompre les données stockées dans un système de persistance par fichiers
- Le SGBD s'assure du maintien de données cohérentes et met en place un mécanisme de récupérations grâce au journal des transactions.
- Permet de revenir à un état cohérent suite à une panne.

La gestion transactionnelle

- Dans un fichier, toute insertion, modification ou suppression est définitive (ou repose sur la dernière sauvegarde réalisée)
- Un SGBD met en place un mécanisme sophistiqué de gestion de transactions.
- **Une transaction** est un ensemble d'opération de manipulations de données ayant un objectif défini.
- Le SGBD garantit la qualité **ACID** de ces transactions :
 - ✓ **A** : Atomacité
 - ✓ **C** : Cohérence
 - ✓ **I** : Isolation
 - ✓ **D** : Durabilité

Quand ne PAS utiliser un SGBD

- Application simple, données peu redondantes et peu variables (ex: une seule entité)
 - Attention au futur!
- Applications en temps réel
 - (ex: analyse de signaux, jeux vidéo, quoique...)
- Systèmes embarqués avec espace disque/mémoire restreint
- Utilisateur unique
- Requêtes sur objets complexes (image, vidéo, ...)



PRINCIPALES ARCHITECTURES

Architecture ?

- Une **architecture** établit la subdivision d'un système ou d'un programme en composants et décrit la relation entre les composants.
- Les SGBDs sont souvent un élément central d'un système informatique.
- Nous décrivons ici les architectures typiques, et le positionnement du SGBD dans l'architecture.

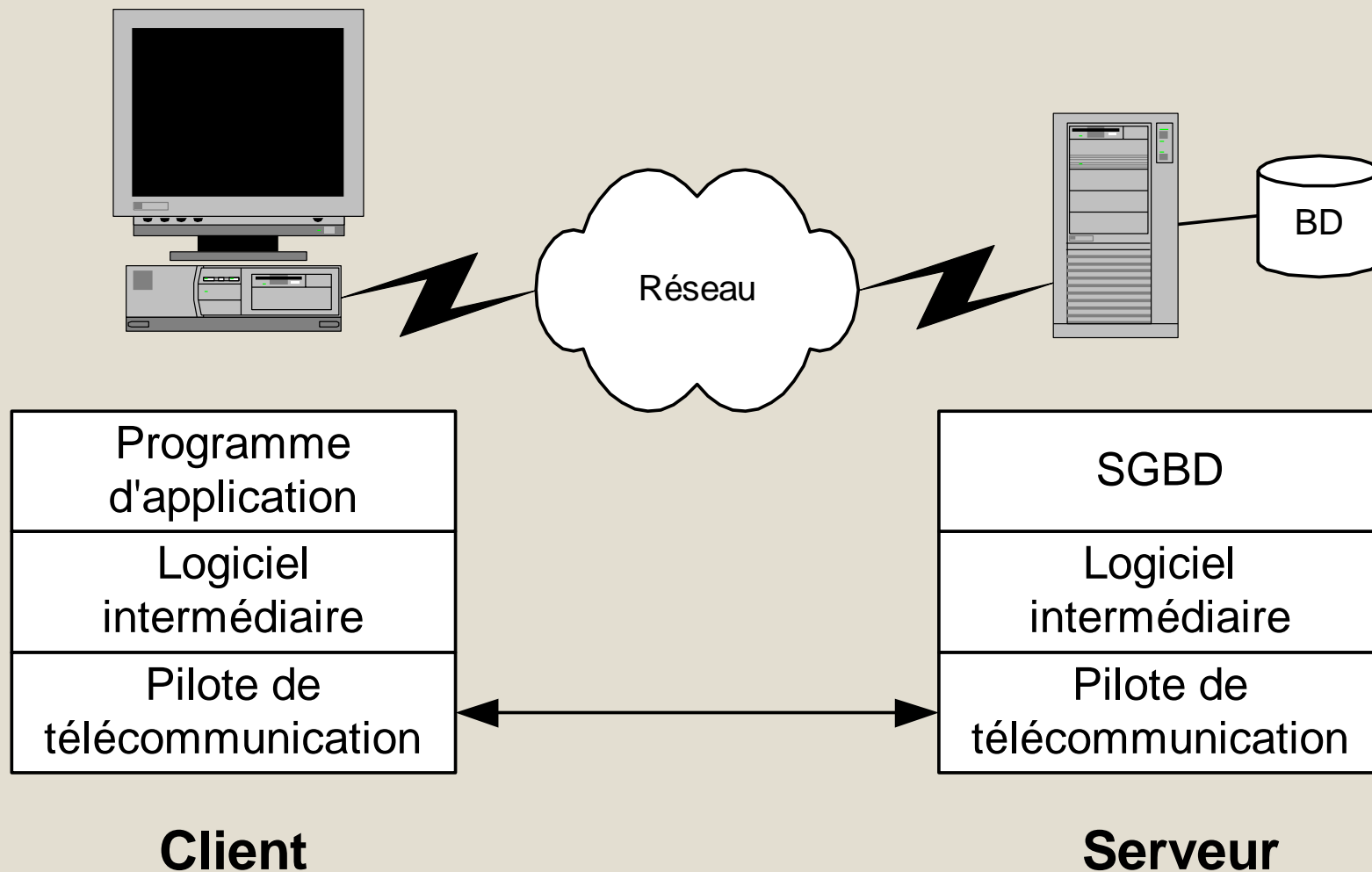
Architecture centralisée

- Architecture simple composée d'un seul nœud (machine)
- Le SGBD réside sur la même machine que le/les programmes qui y accèdent
- L'accès à la base de données se fait en utilisant les API de la base de données, sans passer par la couche réseau.
- Généralement des applications de faible envergure et les premiers systèmes.

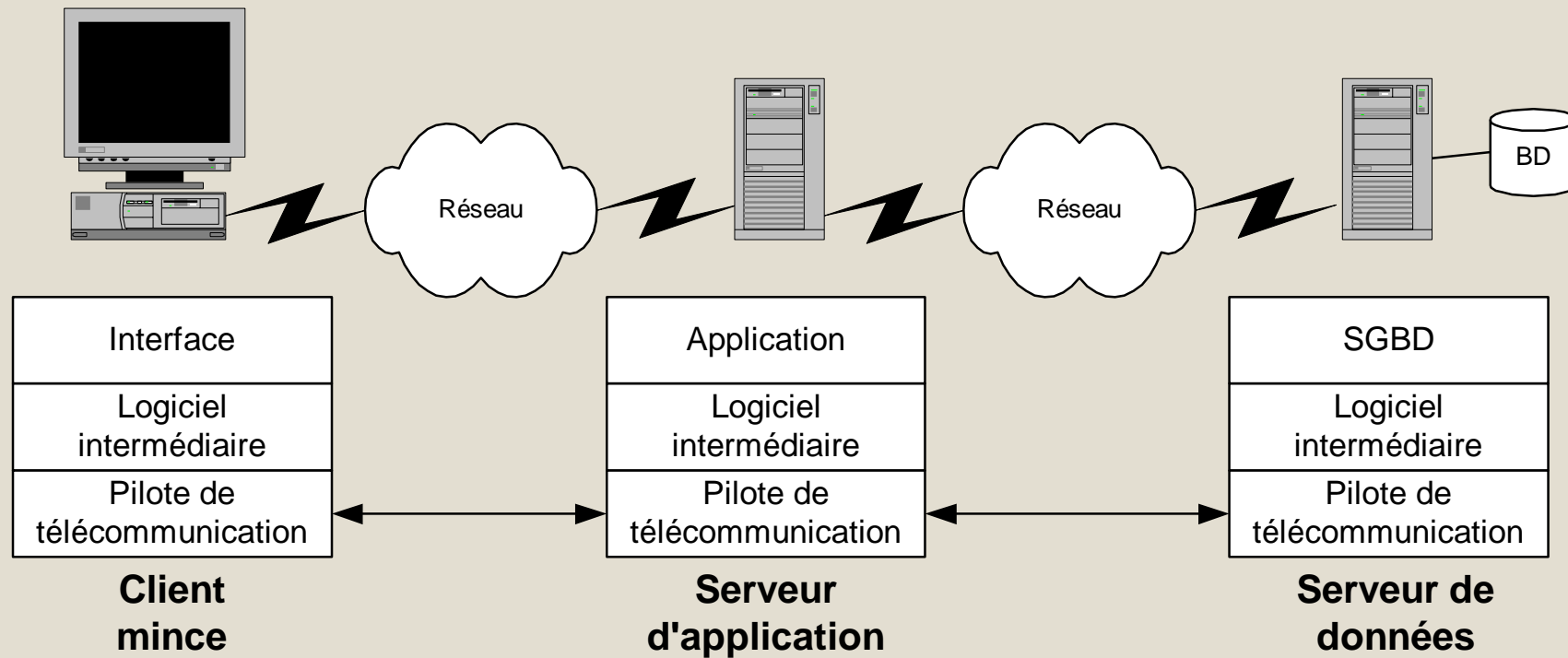
Architectures décentralisées

- Le SGBD (et d'autres composantes de l'architecture) sont réparties sur différents nœuds (machines) qui communiquent à travers le réseau.
- Parmi ces architectures :
 - ❖ Client serveur
 - ❖ Applications n-niveaux
 - ❖ Base de données répartie
 - ❖ Bus d'entreprise
 - ❖ Entrepôt de données

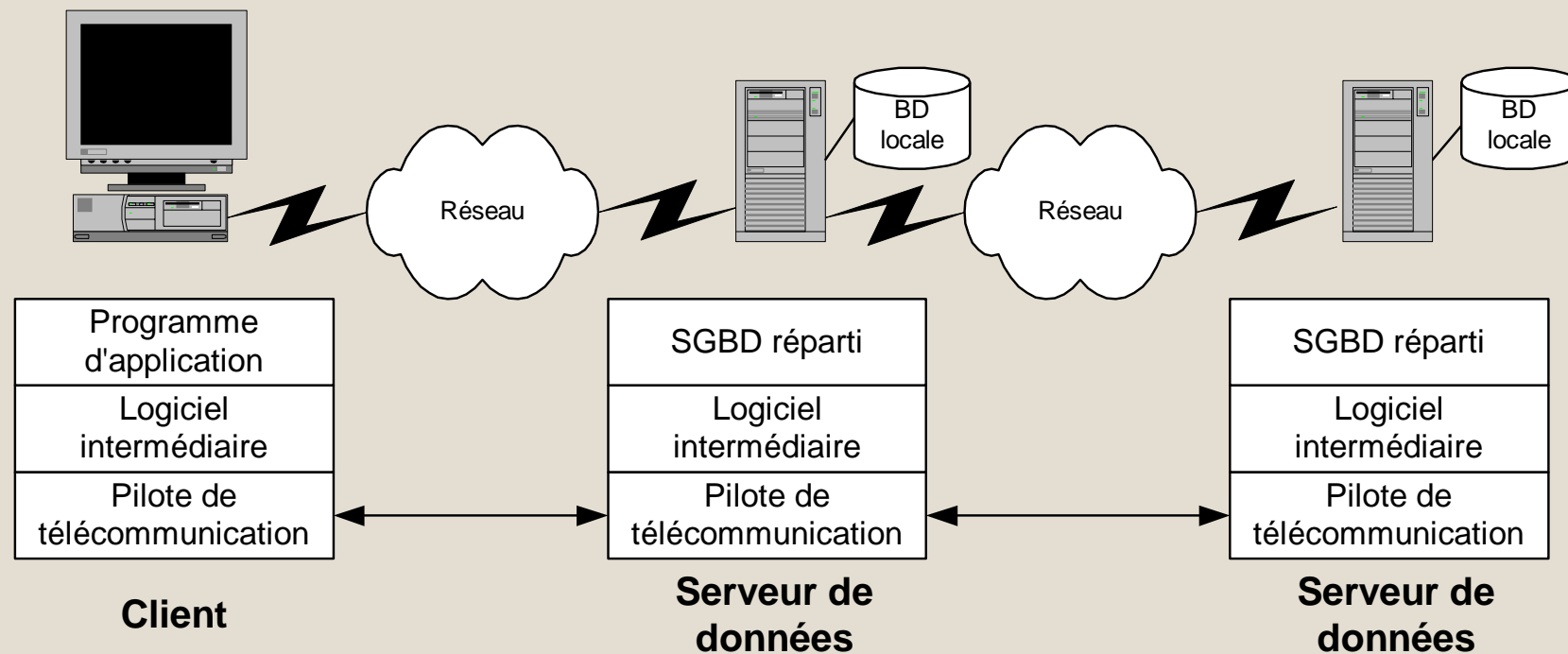
Client-serveur



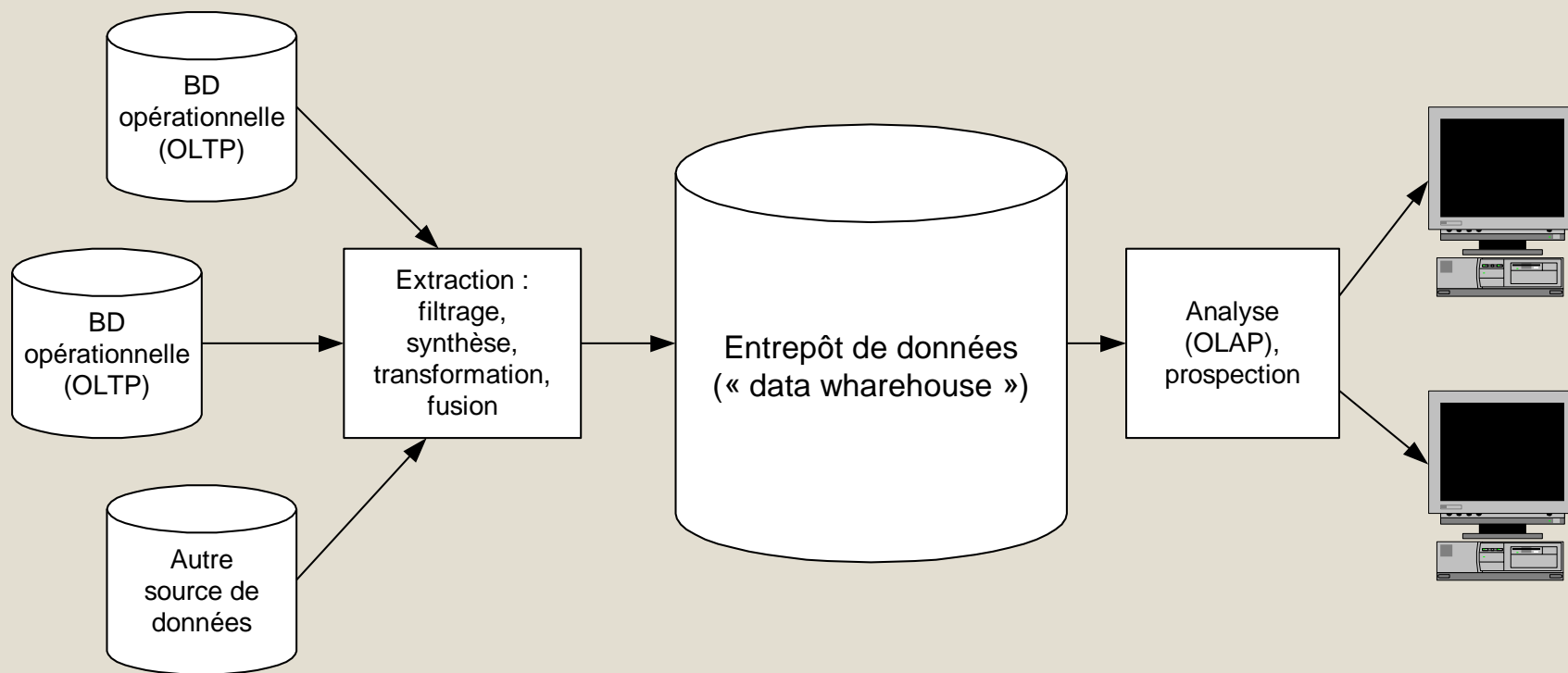
Architecture n-niveaux



SGBD Réparti



Entrepôt de données





EXEMPLE INTRODUCTIF

Banque sans-Intérêts

Table <i>Client</i>			
noClient	nomClient	adresseClient	noTéléphone
10	Hugh Paycheck	Ottawa	(999)999-9999
20	Dollard Cash	Montréal	(888)888-8888
30	Ye San Le Su	Montréal	(777)777-7777

Table <i>Compte</i>			
noCompte	solde	dateOuverture	noClient
100	1000.00	5/05/1999	10
200	2000.00	10/10/1999	20
300	1000.00	10/10/1999	10
400	5.00	20/7/2000	30
600	10.00	15/10/2000	30

Table <i>Prêt</i>					
noPrêt	montantPrêt	dateDébut	tauxIntérêt	fréquence Paiement	noClient
1000	10000.00	10/6/2000	10	12	10
2000	20000.00	20/7/2000	12	52	30
3000	5000.00	15/8/2000	12	12	10

Schéma de la base de données (LDD)

```
CREATE TABLE Client
```

```
(noClient          INTEGER PRIMARY KEY,  
 nomClient         VARCHAR(15),  
 adresseClient     VARCHAR(20),  
 noTéléphone       VARCHAR(15))
```

```
CREATE TABLE Compte
```

```
(noCompte          INTEGER PRIMARY KEY,  
 solde             DECIMAL(10,2) CHECK (solde >= 0),  
 dateOuverture     DATE,  
 noClient          INTEGER REFERENCES Client)
```

```
CREATE TABLE Prêt
```

```
(noPrêt            INTEGER PRIMARY KEY,  
 montantPrêt       DECIMAL(10,2),  
 dateDébut         DATE,  
 tauxIntérêt       DECIMAL(8,2),  
 fréquencePaiement INTEGER,  
 noClient          INTEGER REFERENCES Client)
```

Création du schéma

```
SQL> CREATE TABLE Client
2  (noClient      INTEGER PRIMARY KEY,
3   nomClient     VARCHAR(18),
4   adresseClient VARCHAR(20),
5   noTéléphone   VARCHAR(15))
6  /
```

Table created.

```
SQL> CREATE TABLE Compte
2  (noCompte      INTEGER PRIMARY KEY,
3   solde         DECIMAL(10,2) CHECK (solde >= 0),
4   dateOuverture DATE,
5   noClient      INTEGER REFERENCES Client)
6  /
```

Table created.

```
SQL> CREATE TABLE Prêt
2  (noPrêt       INTEGER PRIMARY KEY,
3   montantPrêt  DECIMAL(10,2),
4   dateDébut    DATE,
5   tauxIntérêt  DECIMAL(8,2),
6   fréquencePaiement INTEGER,
7   noClient     INTEGER REFERENCES Client)
8  /
```

Table created.

Manipulation de données

- Ajout (insertion) d'un nouveau client :

```
SQL> INSERT INTO Client
2     VALUES(10,'Luc Sansom','Ottawa','(999)999-9999')
3     /

1 row created.
```

- Obtention de la liste de clients

```
SQL> SELECT *
2     FROM Client
3     /
```

NOCLIENT	NOMCLIENT	ADRESSECLIENT	NOTÉLÉPHONE
10	Luc Sansom	Ottawa	(999) 999-9999

Transactions

```
-- Session parallèle avec Oracle (multiversion) :
```

```
SQL> SELECT *  
  2 FROM Client  
  3 /
```

```
no rows selected
```

```
SQL> COMMIT  
  2 /
```

```
Commit complete.
```

```
Session parallèle :
```

```
SQL> SELECT *  
  2 FROM Client  
  3 /
```

NOCLIENT	NOMCLIENT	ADRESSECLIENT	NOTÉLÉPHONE
10	Luc Sansom	Ottawa	(999) 999-9999

Insertion par lots (une transaction)

45

```
SQL> INSERT INTO Client
  2   VALUES(20,'Dollard Tremblay','Montréal','(888)888-8888'
  3   /

1 row created.

SQL> INSERT INTO Client
  2   VALUES(30,'Lin Bô','Montréal','(777)777-7777')
  3   /

1 row created.

SQL> ALTER SESSION SET NLS_DATE_FORMAT = 'DD/MM/YYYY'
  2   /

Session altered.

SQL> INSERT INTO Compte
  2   VALUES(100,1000.0,'5/5/1999',10)
  3   /

1 row created.

...

SQL> INSERT INTO Prêt
  2   VALUES(3000,5000,'15/8/2000',12,12,10)
  3   /

1 row created.

SQL> COMMIT
  2   /

Commit complete.
```


Contrainte d'intégrité

```
SQL> INSERT INTO Client
      2     VALUES(10,'Jean Leconte','Montréal','(666) 666-6666')
      3  /
INSERT INTO Client
*
ERROR at line 1:
ORA-00001: unique constraint (IDUTIL1.SYS_C001737) violated
```

Interrogation des données (SELECT)

```
SQL> SELECT  noCompte, solde
2  FROM    Compte
3  WHERE   noClient = 10
4  /
```

NOCOMPTE	SOLDE
100	1000
300	1000

noCompte	solde	dateOuverture	noClient
100	1000.00	5/05/1999	10
200	2000.00	10/10/1999	20
300	1000.00	10/10/1999	10
400	5.00	20/7/2000	30
600	10.00	15/10/2000	30

```
SELECT    noCompte, solde
FROM      Compte
WHERE     noClient = 10
```

noCompte	solde
100	1000.00
300	1000.00

Interrogation depuis une application JAVA

48

```
// Exemple de programme JAVA qui utilise le pilote JDBC thin d'Oracle
// pour effectuer un SELECT et itérer sur les lignes du résultat

// Il faut importer le paquetage java.sql pour utiliser JDBC
package ExemplesJDBC;
import java.sql.*;
import java.math.BigDecimal;

class ExempleSelectCompte
{
    public static void main (String args [])
        throws SQLException, ClassNotFoundException, java.io.IOException
    {
        // Charger le pilote JDBC d'Oracle
        Class.forName ("oracle.jdbc.driver.OracleDriver");

        // Connection à une BD avec un pilote thin
        Connection uneConnection =

DriverManager.getConnection("jdbc:oracle:thin:@127.0.0.1:1521:ora817i",
"idouill", "oracle");

        // Création d'un énoncé associé à la Connection
        Statement unEnoncéSQL = uneConnection.createStatement ();

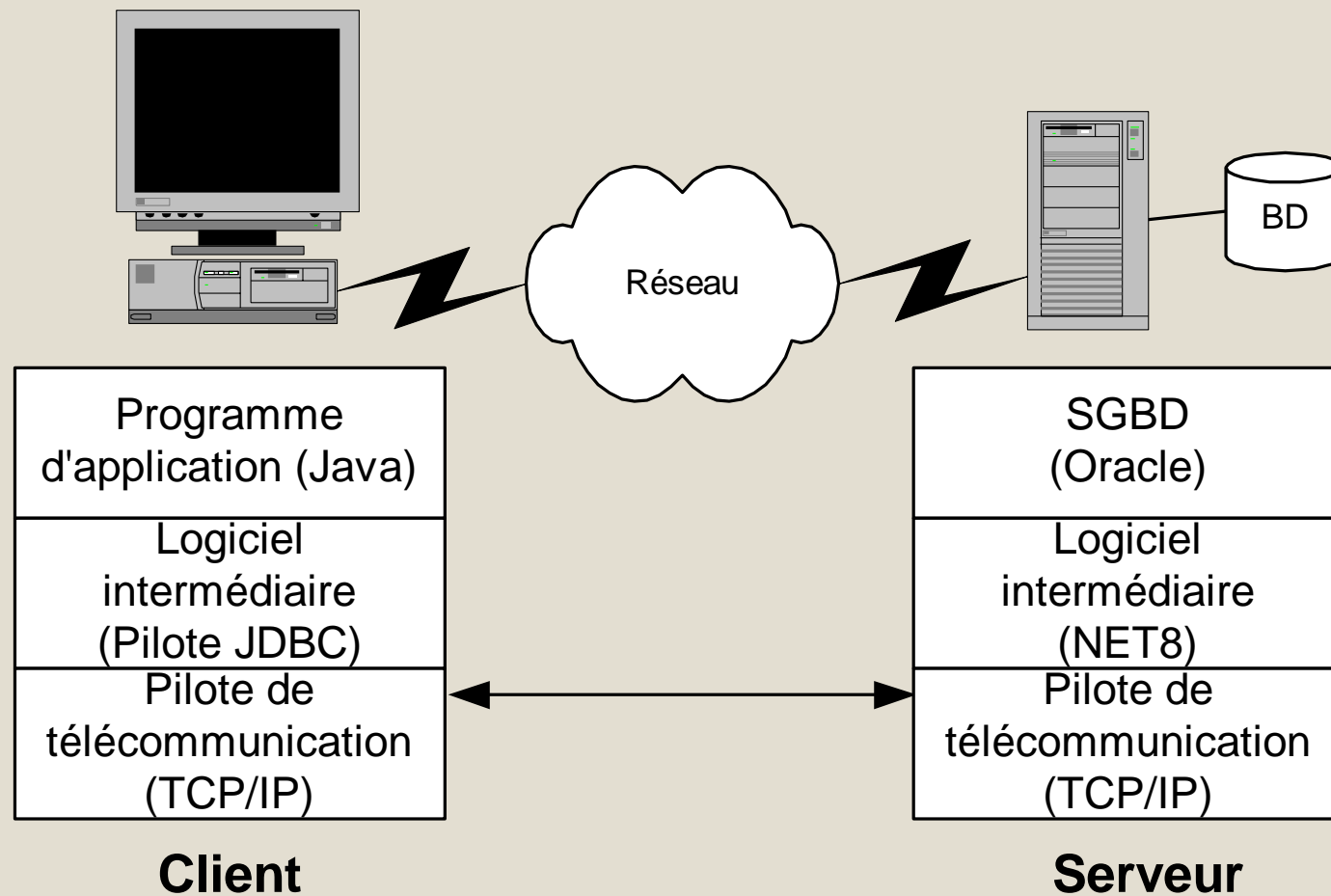
        // Exécution d'un SELECT
        // Le code du SELECT est passé en paramètre sous forme d'un String
        ResultSet resultatSelect = unEnoncéSQL.executeQuery
            ("SELECT noCompte, solde FROM Compte WHERE noClient = 10");

        // Itérer sur les lignes du résultat du SELECT et extraire les valeurs
        // des colonnes dans des variables JAVA
        while (resultatSelect.next ()){
            int noCompte = resultatSelect.getInt ("noCompte");
            BigDecimal solde = resultatSelect.getBigDecimal ("solde");

            System.out.println ("Numéro du compte:" + noCompte);
            System.out.println ("Solde:" + solde);
        }

        // Fermeture de l'énoncé et de la connexion
        unEnoncéSQL.close();
        uneConnection.close();
    }
}
```

Pilote JDBC



Select – Plusieurs tables

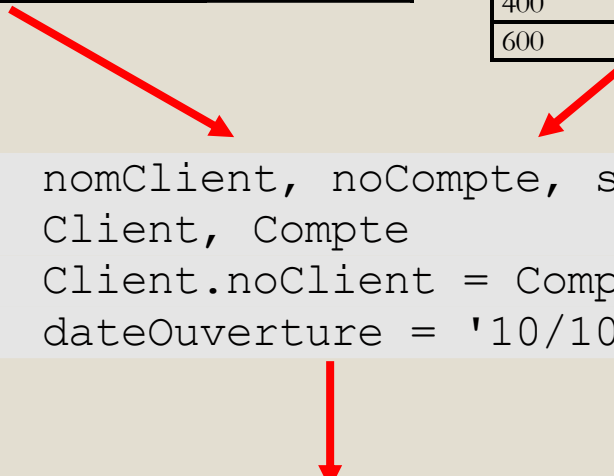
```
SQL> SELECT  nomClient, noCompte, solde
      2  FROM    Client, Compte
      3  WHERE   Client.noClient = Compte.noClient AND
      4          dateOuverture = '10/10/1999'
      5  /
```

NOMCLIENT	NOCOMPTE	SOLDE
-----	-----	-----
Dollard Tremblay	200	2000
Luc Sansom	300	1000

Select – Plusieurs tables

noClient	nomClient	adresseClient	noTéléphone
10	Hugh Paycheck	Ottawa	(999)999-9999
20	Dollard Cash	Montréal	(888)888-8888
30	Ye San Le Su	Montréal	(777)777-7777

noCompte	solde	dateOuverture	noClient
100	1000.00	5/05/1999	10
200	2000.00	10/10/1999	20
300	1000.00	10/10/1999	10
400	5.00	20/7/2000	30
600	10.00	15/10/2000	30



```
SELECT nomClient, noCompte, solde
FROM Client, Compte
WHERE Client.noClient = Compte.noClient AND
dateOuverture = '10/10/1999'
```

nomClient	noCompte	solde
Dollard Cash	200	2000.00
Hugh Paycheck	300	1000.00

Mise à jour des données

```
SQL> UPDATE Compte
2  SET solde = solde - 100
3  WHERE noCompte = 100
4  /
```

1 row updated.

```
SQL> UPDATE Compte
2  SET solde = solde + 100
3  WHERE noCompte = 300
4  /
```

1 row updated.

```
SQL> COMMIT
2  /
```

Commit complete.

```
SQL> SELECT  noCompte, solde
2  FROM      Compte
3  WHERE     noClient = 10
4  /
```

NOCOMPTE	SOLDE
100	900
300	1100

Suppression de données

```
SQL> DELETE FROM Compte WHERE noCompte = 100
2 /
```

1 row deleted.

```
SQL> COMMIT
2 /
```

Commit complete.

```
SQL> SELECT * FROM Compte
2 /
```

NOCOMPTE	SOLDE	DATEOUVERT	NOCLIENT
-----	-----	-----	-----
200	2200	10/10/1999	20
300	1000	10/10/1999	10
400	5	20/07/2000	30
600	10	15/10/2000	30

Dictionnaire de données

```
SQL> SELECT TABLE_NAME  
2 FROM USER_TABLES  
3 /
```

TABLE_NAME

CLIENT

COMPTE

PRÊT

```
SQL> SELECT COLUMN_NAME  
2 FROM USER_TAB_COLUMNS  
3 WHERE TABLE_NAME = 'CLIENT'  
4 /
```

COLUMN_NAME

NOCLIENT

NOMCLIENT

ADRESSECLIENT

NOTÉLÉPHONE

Postgresql (avec pgadmin4)

- Téléchargement de PostgreSQL (inclut pgAdmin4)
 - <https://www.postgresql.org/download/>
- Comment utiliser pgAdmin4
 - <https://www.usherbrooke.ca/informatique/intranet/ressources-et-documentation/logiciels-services-outils/postgresql/>
 - Hôte: bd-info3.dinf.usherbrooke.ca
 - Port: 5432

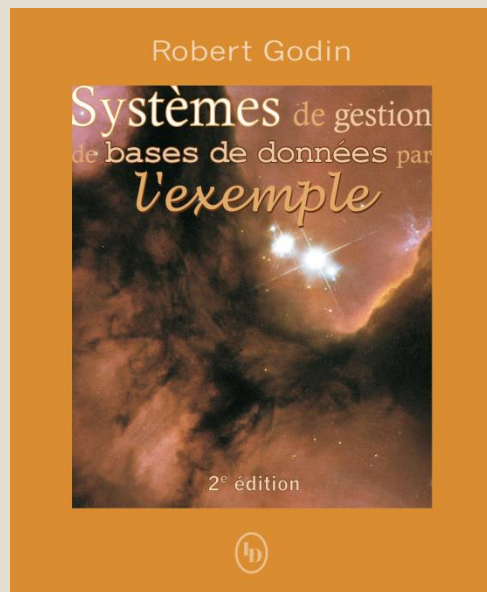
Quelques erreurs

- Quelques particularités du type SERIAL
 - <https://stackoverflow.com/questions/787722/postgresql-autoincrement>
- Tables inexistantes à cause des guillemets:
 - <https://stackoverflow.com/questions/6331504/omitting-the-double-quote-to-do-query-on-postgresql>
 - Solution: pas de majuscules dans vos noms de table

Vocabulaire

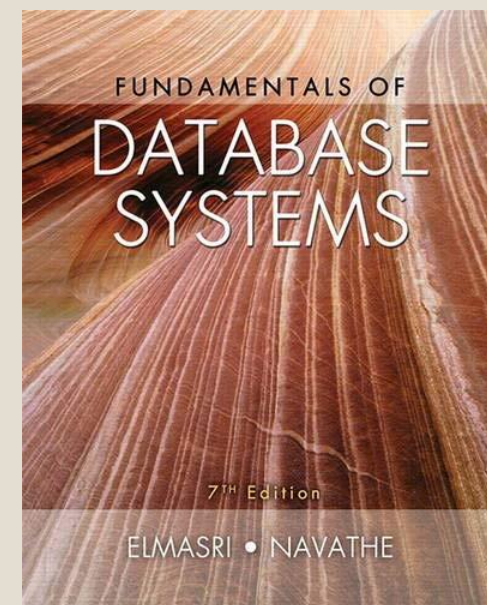
- **BD** - Base de données
- **LDD** - Langage de définition des données
- **LMD** - Langage de manipulation des données
- **LCD** - Langage de contrôle des données
- **SGBD** - Système de gestion de bases de données
- **Catalogue de BD** - Description de la structure de la BD (schéma)
- **Requête** - Demande d'un sous-ensemble de données satisfaisant des contraintes
- **Transaction** - Ensemble d'opérations à effectuer sur la BD
- **SQL** - Structured Query Language (langage de requête structurée)
- **MCD** - Modèle conceptuel des données

RÉFÉRENCES



Godin, R. (2012).
*Systèmes de gestion de
bases de données par
l'exemple*. 3^{ième} édition,
Montréal, Canada,
Loze-Dion.

Elmasri, R. et Navathe, S. (2016)
*Fundamentals of Database
Systems*, 7th Edition
Pearson



La SEMAINE PROCHAINE

Modèle conceptuel

« Le modèle conceptuel des données (**MCD**) a pour but d'écrire de façon formelle les données qui seront utilisées par le système d'information. Il s'agit donc d'une représentation des données, facilement compréhensible, permettant de décrire le système d'information à l'aide d'entités. » MERISE