Panorama des Bases de Données

Didier Donsez

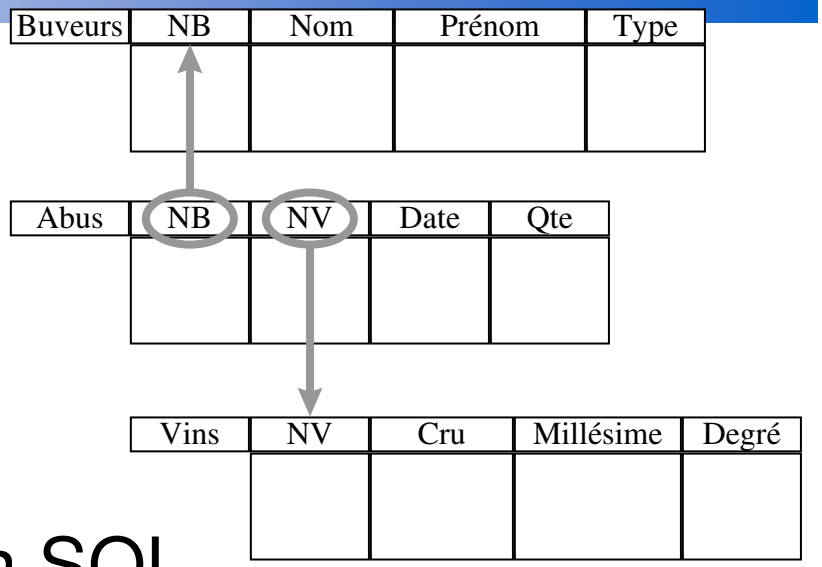
Université Joseph Fourier (Grenoble 1)

IMA IMAG/LSR/ADELE

Didier.Donsez@imag.fr

Le Modèle Relationnel

- Un modèle de données
  - Simple
    - Relation (Table)
    - Tuple (Ligne)
    - Attribut (Colonne)

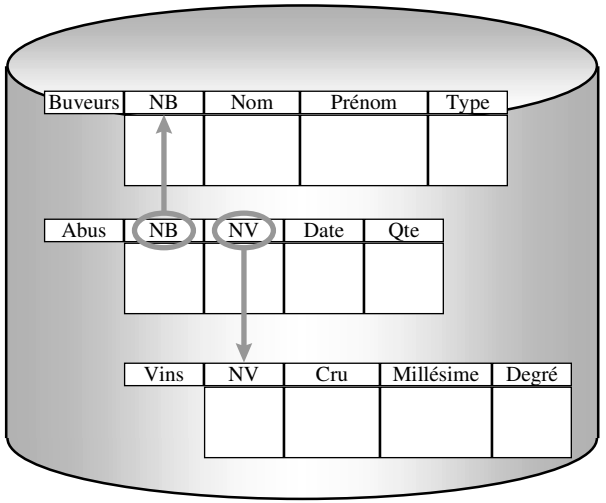


- Un langage de manipulation SQL

```
SELECT  BUVEURS.NOM, SUM(ABUS.QTE)
FROM    BUVEURS, ABUS
WHERE   BUVEURS.NB=ABUS.NB
        AND BUVEURS.VILLE='Valenciennes'
GROUP BY BUVEURS.NOM
```

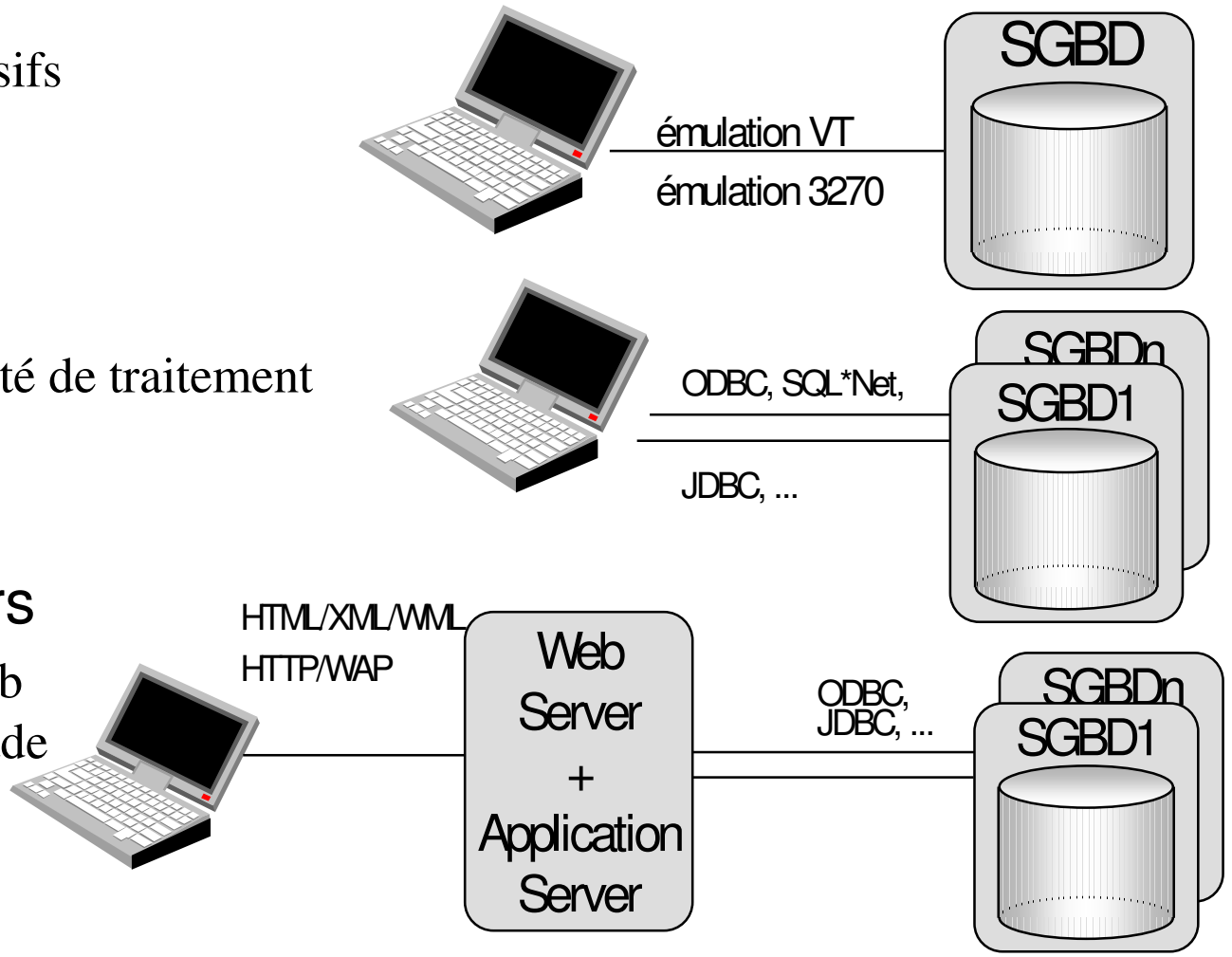
les SGBDs Relationnels

- Des produits murs
  - ORACLE
  - SYBASE
  - INFORMIX
  - IBM DB/2
  - OPEN INGRES (CA)
  - MICROSOFT SQL SERVER
- Des produits Open Sources
  - MySQL
  - PostGres
  - HyperSonic
  - …



Le modèle d'accès aux SGBDs (Relationnels)

- Centralisé
  - Terminaux passifs
- Client Serveur
  - PC avec capacité de traitement
- Web Multi-Tiers
  - Navigateur Web
    Terminal nomade

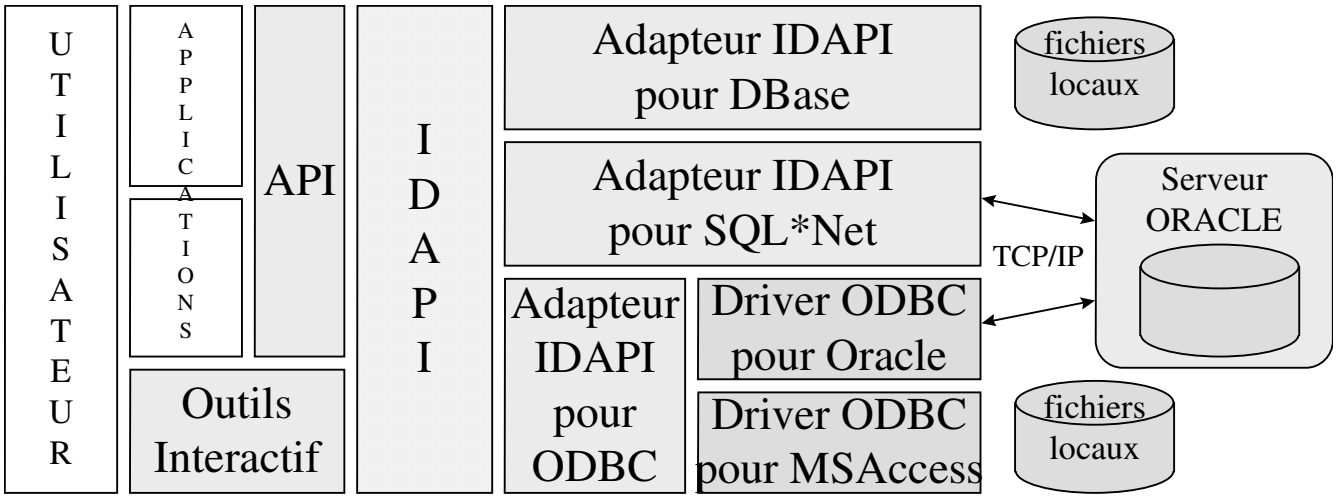


Programmation Procédurale et SQL

- SQL : langage déclaratif
  - optimisation de la requête par le SGBD
- Mais besoin de programme procéduraux
  - boucle, test, affichage, enchainement, …
  - curseur sur une table résultat
- Solutions
  - Middleware
    - ODBC, JDBC
  - Embedded SQL
    - C, Java
  - P/SQL

Les Middlewares SQL/CLI

- Ordre SQL dans un langage général
  - pas de précompilation
  - indépendance aux SGBDs (et autres sources de données)
- Normalisation SQL/CLI (Call Level Interface)
  - ODBC Open DataBase Connectivity (MicroSoft) *SQL/CLI*
  - JDBC Java DataBase Connectivity (JavaSoft) *SQL/CLI*
  - IDAPI Integrated Database Application Interface (Borland)
  - ...

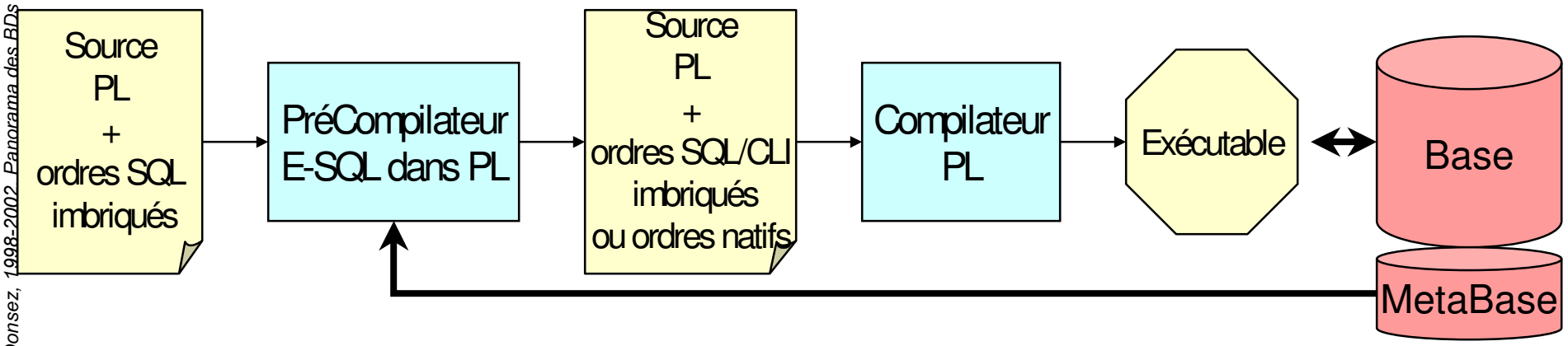


Exemple JDBC
Java DataBase Connectivity

```
class Employe {
   public static void main (String args [] )
                throws SQLException, ClassNotFoundException {
   Class.forName ("oracle.jdbc.driver.OracleDriver");
   String dburl = "jdbc:oracle:oci8:@";
   Connection conn = DriverManager.getConnection(dburl, "toto", "passemot");
   Statement stmt = con.createStatement();
   ResultSet rs = stmt.executeQuery
                ("SELECT numemp, name, salary FROM EMPLOYE");
   while (rs.next()) {
      String s = rs.getString(2); float f = rs.getFloat("salary");
     if(rs.wasNull())      System.out.println (s + " n"a pas de salaire");
                else       System.out.println (s + " gagne "+ f +" $");
   } rs.close();
} catch(Exception e) {
   e.printStackTrace(); } }
```

Embedded SQL

- Syntaxe plus concise que SQL/CLI
- Analyse statique
  - Contrôle de la Syntaxe et du Typage
  - Typage curseur dépendant de la Métabase
- Précompilation : Pro*C d 'Oracle, JSQL



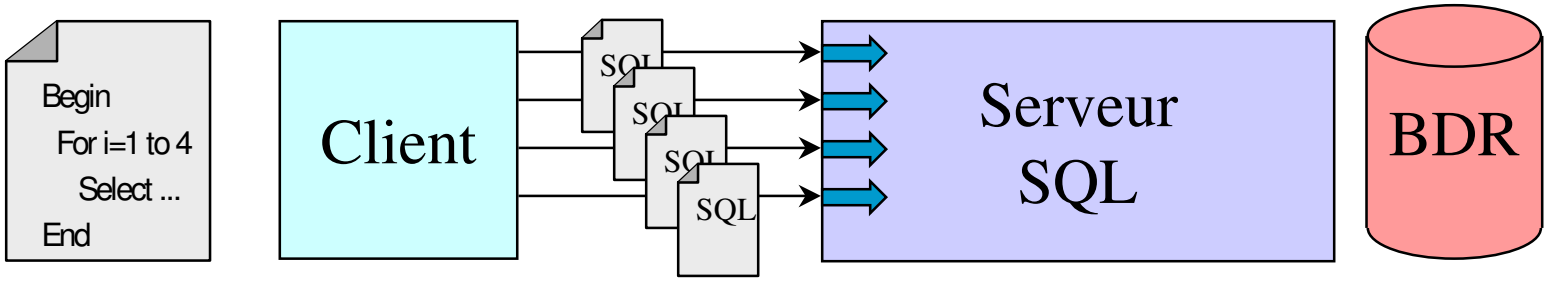
Exemple de Curseur en Pro*C

```
EXEC SQL BEGIN DECLARE SECTION;
   char nom[21];      float salaire;
EXEC SQL END DECLARE SECTION;
. . .
EXEC SQL DECLARE c CURSOR FOR
        SELECT name, salary FROM Employe WHERE salary > 10000;
EXEC SQL OPEN CURSOR c;
while(1) {
   EXEC SQL FETCH c  INTO :nom, :salaire;
   if(NOT FOUND) break else printf ("%s gagne %d $\n", nom, salaire);
}
EXEC SQL CLOSE CURSOR c;
```

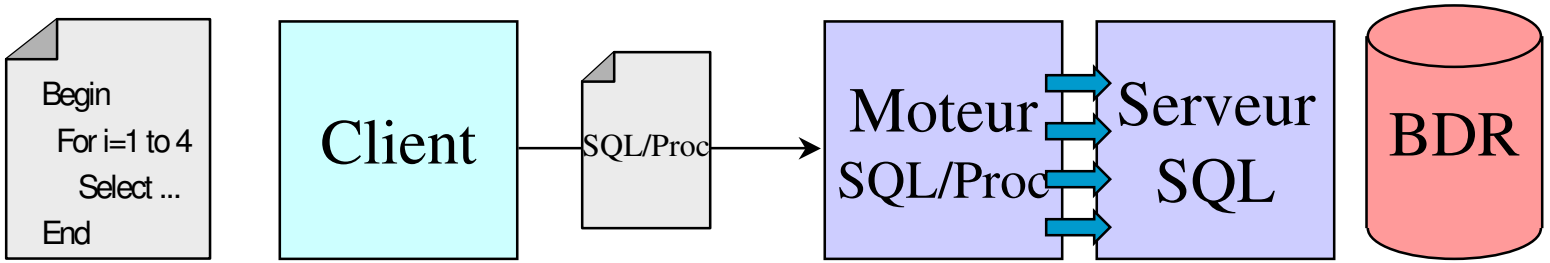
Exemple SQLJ

```
#sql public iterator IterEmp (String, int); // déclaration d 'une classe d 'itérateur
{
        IterEmp iter; // déclaration d 'un objet itérateur
        String nom; int sal; int c:=1;
        #sql iter = { SELECT name, salary FROM Employe };
        while (true) {
            #sql { FETCH :iter INTO :nom, :sal };
            if (iter.endFetch()) break;
            if(c++%2) System.out.println( nom + " est payé " + sal + " $" );
        }
        iter.setRow(1); // se repositionne au premier résultat
        while(iter.next(2)) {// se positionne sur les résultats en position impaire
            System.out.println( iter.name() + " est payé " +iter.salary() + " $" );
        }
}
```

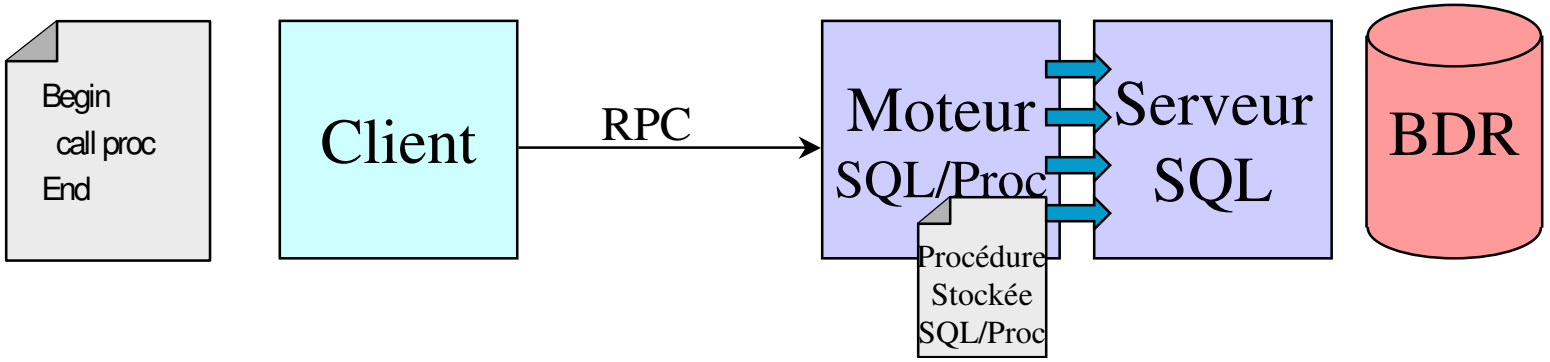
Motivations PL/SQL



- Exécution de la procédure par le serveur



- Appel de procédure stockée par le serveur



Exemple de Curseur en PL/SQL

```
CREATE OR REPLACE PROCEDURE augmentationSalaire(
            seuil                   IN Employe.salary%TYPE,
            augmentation            IN NUMBER(2)
) AS
   sal Employe.salary%TYPE;
   num Employe.numemp%TYPE;
   CURSOR c IS SELECT salary, numemp FROM Employe;
BEGIN
      OPEN c;
      FETCH c INTO sal, num;      -- attention à l'ordre : types compatibles
      WHILE c%NOTFOUND LOOP
            IF sal IS NOT NULL AND sal < seuil THEN
               UDPATE Employe SET salary = salary*(augmentation + 100.0)/100
                  WHERE numemp = num;
            END IF;
            FETCH c INTO sal, num;
      END LOOP;
      CLOSE c;
END;
```

Les Déclencheurs SQL (i)

- Base de Données Active
  - réagit aux changements d ’état de la base de données
- Déclencheur = Evénement-Condition-Action
  - Evénement dans la base
  - Condition
  - Déclenchement d ’une action
- Trigger SQL
  - Evénement
    = INSERT, DELETE, UPDATE dans une relation
  - Action = un ou plusieurs ordres SQL, SQL procédural

Les Déclencheurs SQL (ii)

- Pourquoi faire ?
  - valider les données entrées
  - créer un audit de la base de données
  - dériver des données additionnelles
  - maintenir des règles d ’intégrité complexes
  - implanter des règles de métier
  - supporter des alertes (envoi de e-mails par exemple)
- Gains
  - développement plus rapide
    - les triggers sont stockées dans la base
  - maintien global des règles d ’intégrité

Exemple de Trigger Ordre

Vente(gencod, qte, prix) VolumeAffaire(total,date)

```
CREATE TRIGGER tg_modifVolume AFTER INSERT ON Vente
    DECLARE s number;
    BEGIN
      select sum(prix*qte) into s from Vente;
      insert into VolumeAffaire value(s,current);
    END;

CREATE TRIGGER tg_modifInterdit
    AFTER UPDATE OF prix, qte ON Vente
    BEGIN raise_application_error(-9998, 'Modification interdite '); END;
```

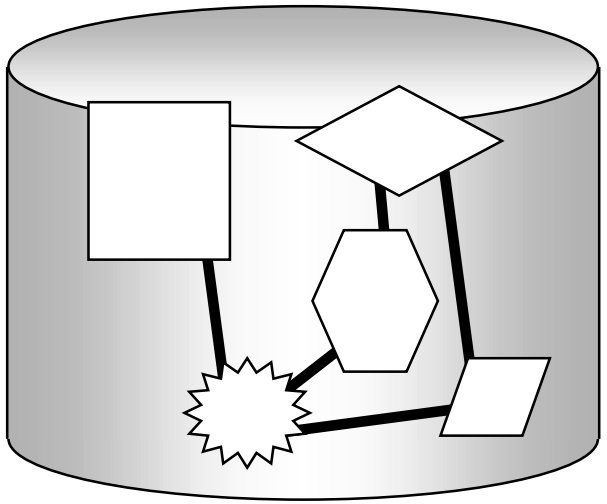
Exemple de Trigger Ligne

Vente(gencod, qte, prix) Stock(gencod, qte)

```
CREATE TRIGGER tg_nouvVente AFTER INSERT ON Vente
FOR EACH ROW
   BEGIN
     if :new.qte > (select qte from Stock where gencod = :new.gencod)
     then     raise_application_error(-9997, ' Stock insuffisant ');
     else     update Stock set qte := Stock.qte - :new.qte
                        where gencod = :new.gencod;
   END;
```

Les concepts objets dans les BD

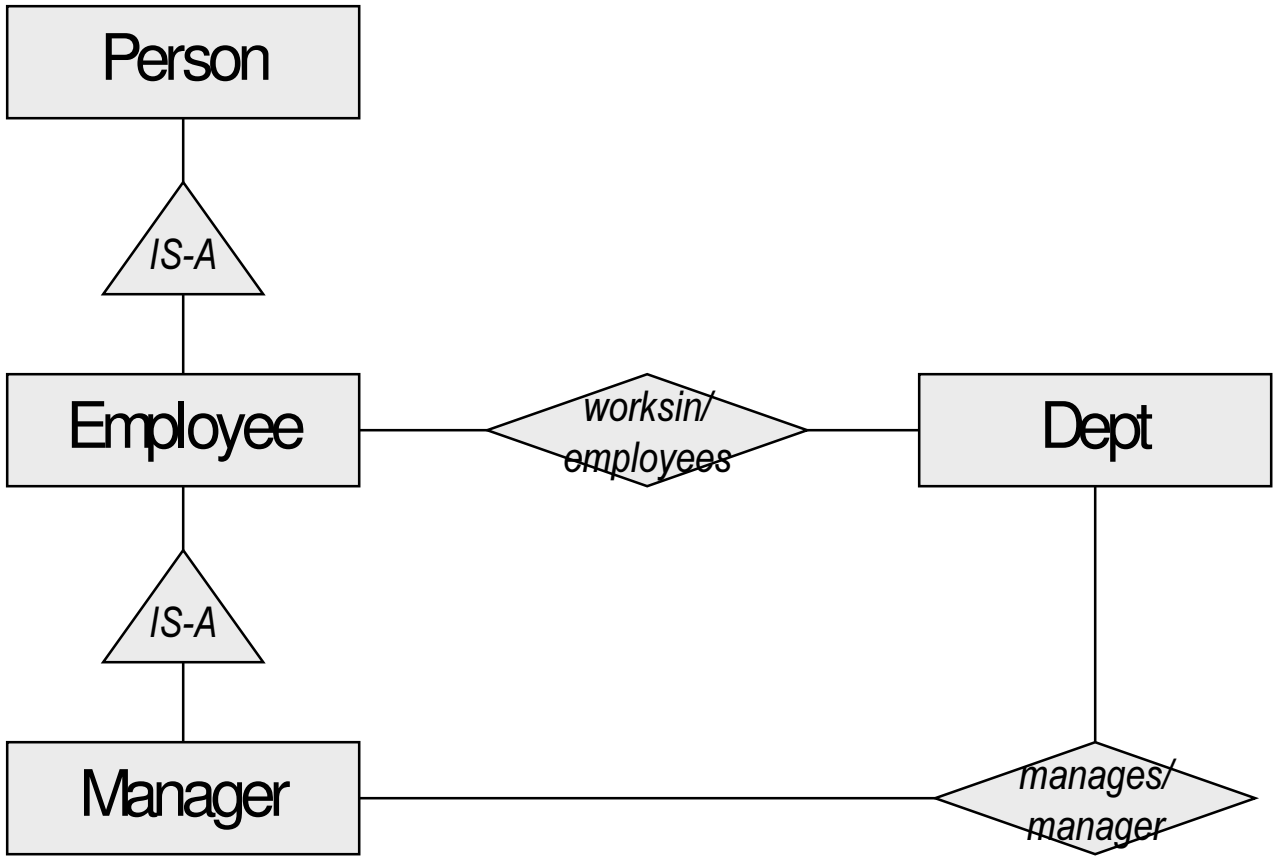
- Concepts Orienté Objet
  - Classe, Objet, Méthode
  - Encapsulation, Héritage
- De nouvelles applications avec des données complexes
  - Médicale, CAO / DAO, SIG …
- Une autre charge de travail
  - reflète l 'activité de conception
    - Transactions longues
    - Workflow, Coopération



les SGBDs Objets Purs (i)

- Né d 'un constat
  - SQL inadéquat aux types complexes et au navigationnel
- Normalisation ODMG : Object Data Management Group
  - Définition d 'un Modèle de données pivot
    - ODL
  - pour des langages généraux
    - C++, Smalltalk, Java, OQL
- Produits
  - ObjectStore, O2, GemStone, POET, …
- Niche : applications ciblées
  - Médicale, CAO / DAO, SIG ...

Exemple Personne-Employé-Manager



Exemple d ’ODL

```
interface Person {
    attribute string name;
    attribute date_t birthday
    attribute Addr addr;

    int age();
}
```

```
interface Employee : Person {
    attribute int numemp;
    attribute int basesalary;
    relationship Dept worksin
        inverse Dept::employees;
    float salary();
}
```

```
interface Manager : Employee {
    attribute int bonus;
    relationship Dept manages
        inverse Dept::manager;
    float salary();
}
```

Exemple d ’ODL

```
interface Dept {
   attribute string name;
   attribute Addr postaddr;

  relationship Set<Employee> employees
        inverse Employee::worksin
  relationship Manager manager
        inverse Manager::manages

   int allsalaries();
}
```

Exemple de Requêtes OQL

- Extent, Collection

```
SELECT e.name, d.addrpost
FROM Employees e, e.worksin d
WHERE d.name="R&D"
```

- Requête imbriquée

```
SELECT m.name, m.salary()
FROM (   SELECT d.manager FROM Depts d) m
WHERE m.salary > 1000000
```

- Groupage

```
SELECT departement : d.name, massesal : SUM(e.salary)
FROM Depts d, d.employees e
GROUP BY d.name
```

Le Modèle Objet-Relationnel (i)

- Extension du modèle relationnel aux concepts objets
  - Définition de User Defined data Type (UDT)
    - Données + Méthodes, Structuration, Héritage, N1NF Values, Références ...
- Normalisation dans SQL 3
- Intérêt :
  - Systèmes patrimoniaux (Legacy Systems)
    - Conserve la compatibilité des applications relationnelles
  - Evolutivité douce
    - Structurations plus complexes (i.e. riches)
    - Définition d 'Object View sur Base Relationnelle

Le Modèle Objet-Relationnel (ii)

Exemple de type avec des références

- Déclaration d 'un UDT

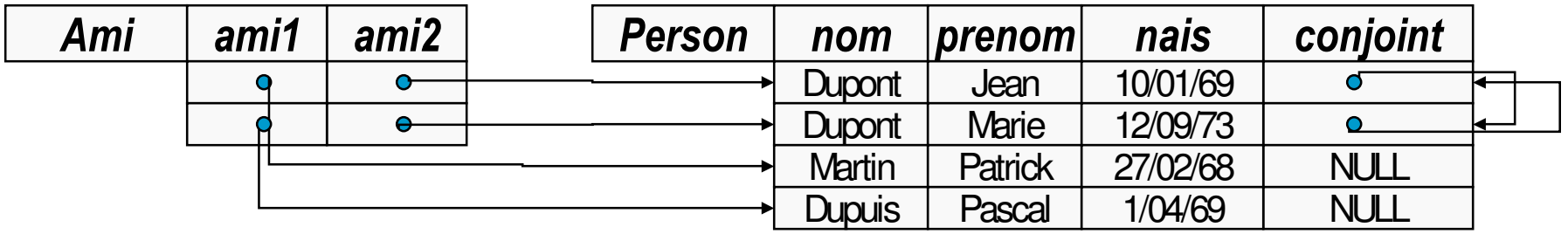
```
CREATE TYPE person_t;
CREATE OR REPLACE BODY TYPE person_t (
    nom VARCHAR(20), prenom VARCHAR(10), nais DATE,
    conjoint            REF      person_t
);
```

- Déclaration d ' une table

```
CREATE TABLE Person OF person_t
                                    (SCOPE FOR conjoint IS Person) ;
CREATE TABLE Ami (
    ami1      REF      person_t,
    ami2      REF      person_t,
    SCOPE FOR ami1 IS Person,
    SCOPE FOR ami2 IS Person
);
```

Le Modèle Objet-Relationnel (iii)

Exemple de déréférenciation




```
SELECT a.ami2->conjoint->nom FROM Ami a
WHERE a.ami1->nom = 'Martin';
```

- équivalent à

```
SELECT x.nom FROM Ami a, Person x, Person y, Person z
WHERE     z.nom = 'Martin'
    AND   REF(z)=a.ami1
    AND   a.ami2=REF(y)
    AND   y.conjoint=REF(x);
```

Les SGBDs Universels

- Capable de supporter tout type de données
  - Texte plein, Texte semi structuré, Image, Vidéo, Données Spatiales, Séries Temporelles, Empreintes Digitales, Séquences génomiques ...
- Supporter c 'est
  - Stocker
  - Manipuler
  - Chercher, Comparer, Retrouver
    - Exemples des empreintes digitales similaires
    - ```
      select S.Nom, I.AffaireId
      from Suspects S, Indices I
      where S.Empreinte Similar I.Empreinte
      ```
- Extension par
  - UDT Objet Relationnel
  - « Data Cartridges » vendus par des tiers
      ```

les SGBDs Multimédia

- Type de données multimédia
  - Image, Son, Vidéo, 3D
- Indexation par le contenu
  - Quel film contient dans la bande sonore la phrase « T 'as beaux yeux, tu sais ! »
  - Données des index : RDF, MPEG7, …
- Stockage et Distribution Temps Réel
  - Vidéo à la Demande (VOD, Near VOD, PreFetch VOD, …)

Les SGBDs Parallèles

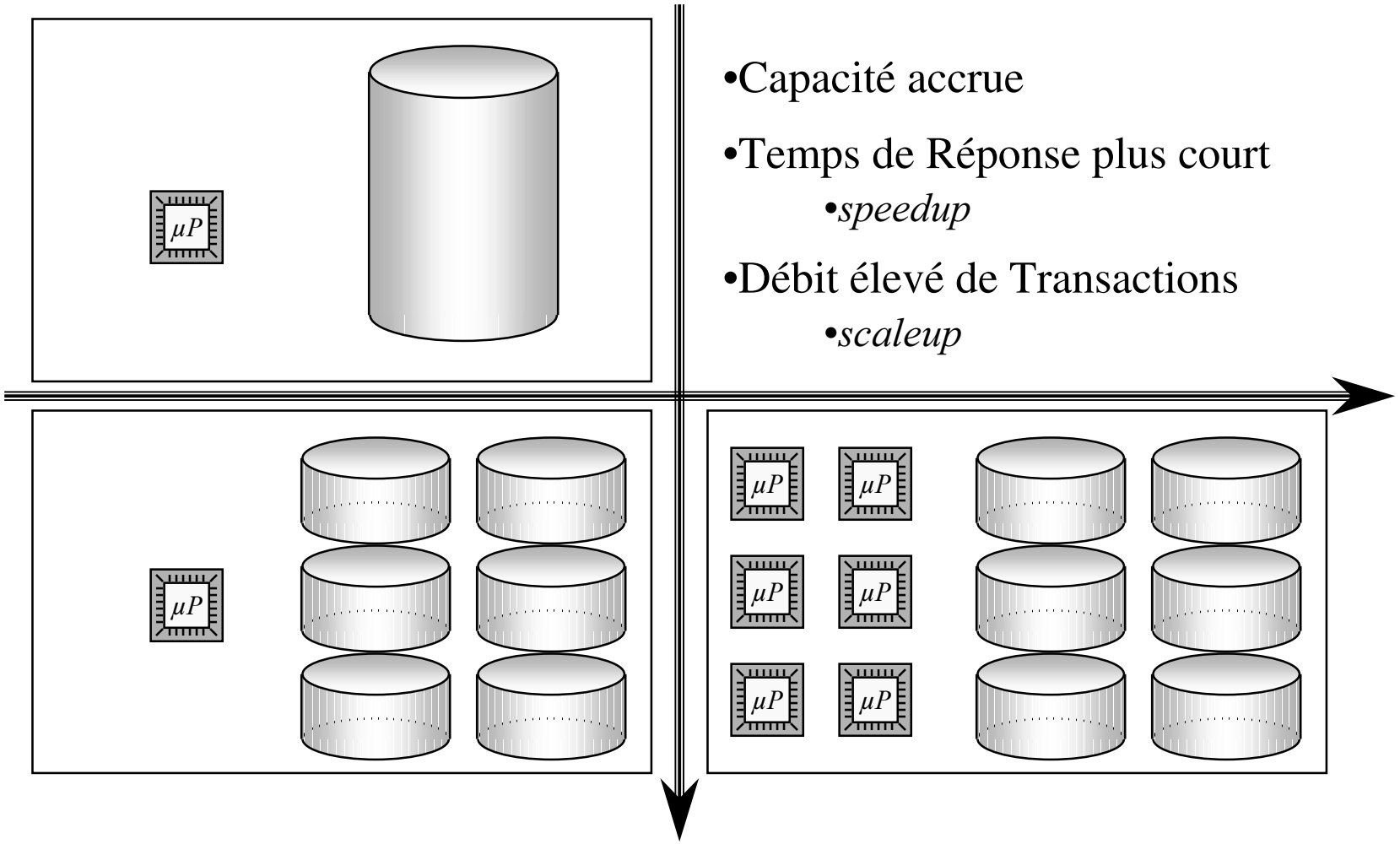
Le problème des Systèmes d 'Information

- Les entreprises dépendent de l'information à jour disponible à temps.
  - augmentation du volume d'information : 30% par an
  - augmentation du volume des transactions : x 10 dans 5 ans
- La charge de travail change :
  - simple OLTP (transactionnel)
  - transactions complexes (e.g., comme support au décisionnel)

Le Besoin :

- des serveurs bases de données qui fournissent à haut débit et avec de bons temps de réponse des charges de travail variées sur des très grosses bases de données.

les SGBDs Parallèles

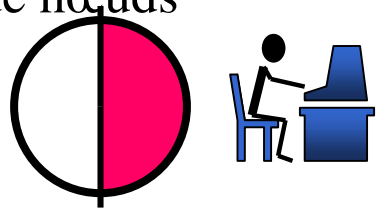


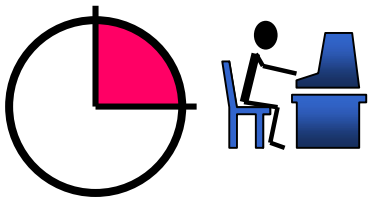
- Capacité accrue
- Temps de Réponse plus court
  - *speedup*
- Débit élevé de Transactions
  - *scaleup*

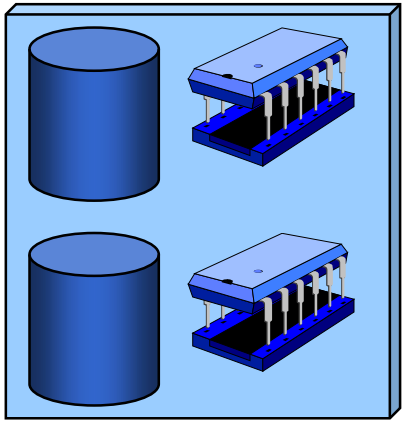
Speed-up et Scale-up

Speed-up

- Diminution du temps de réponse en augmentant le nombre de nœuds

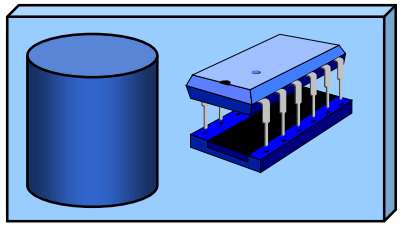


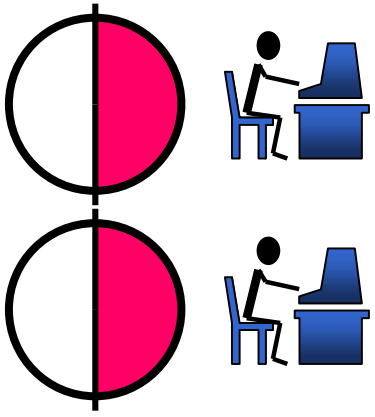


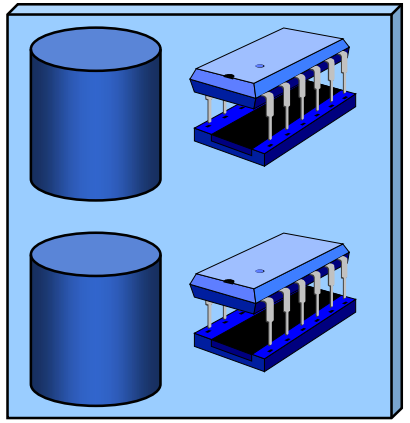


Scale-up

- Accroissement linéaire de la Charge de Travail

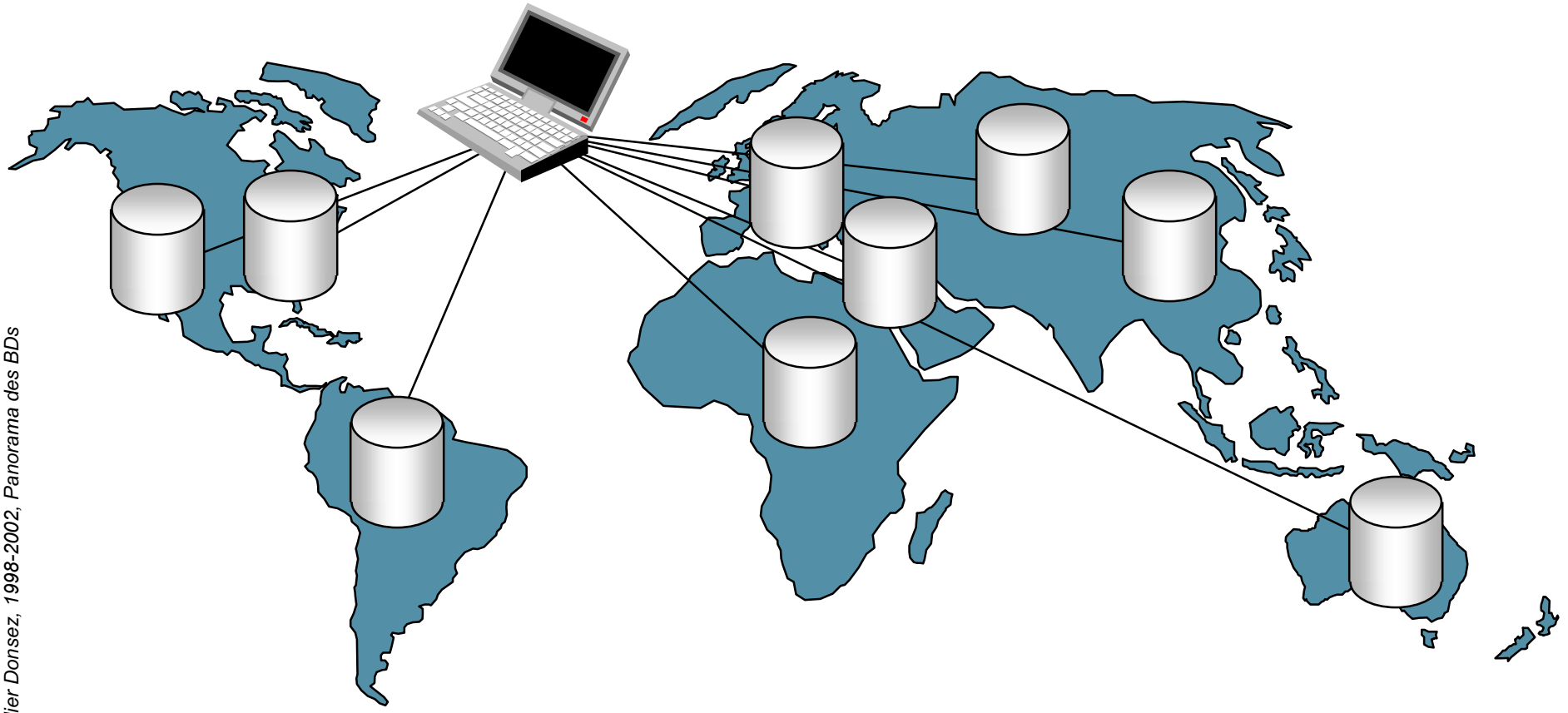






les SGBDs Distribués

- Fragments et réplicas sur N sites distantes



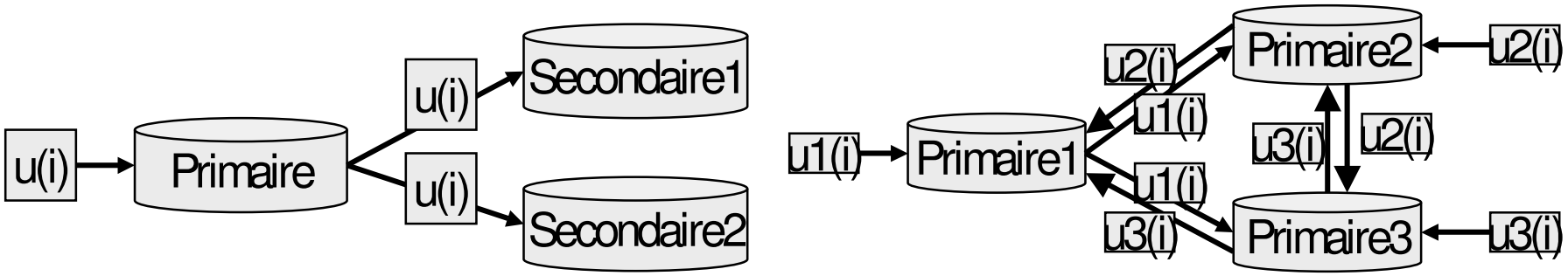
La réplication dans les SGBD

Motivations

- Garantir la disponibilité du Système
  - *Rappelez vous de l 'incendie du Crédit Lyonnais !*
- Performance pour la localité des accès
  - *Imaginez le taux de TVA sur un seul site !!!*

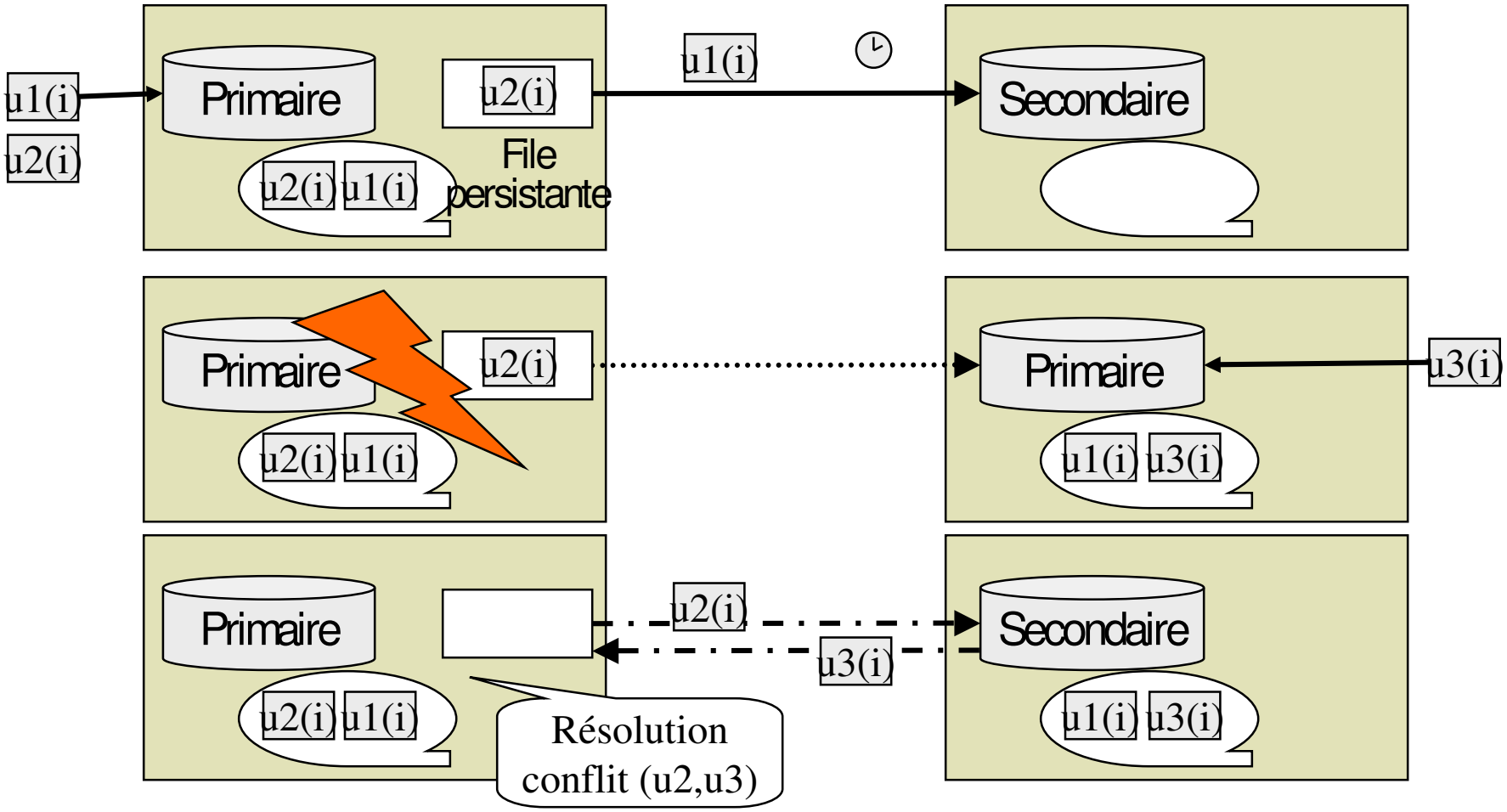
Modes de Réplication

- Asymétrique / Symétrique



Exemple de réplication

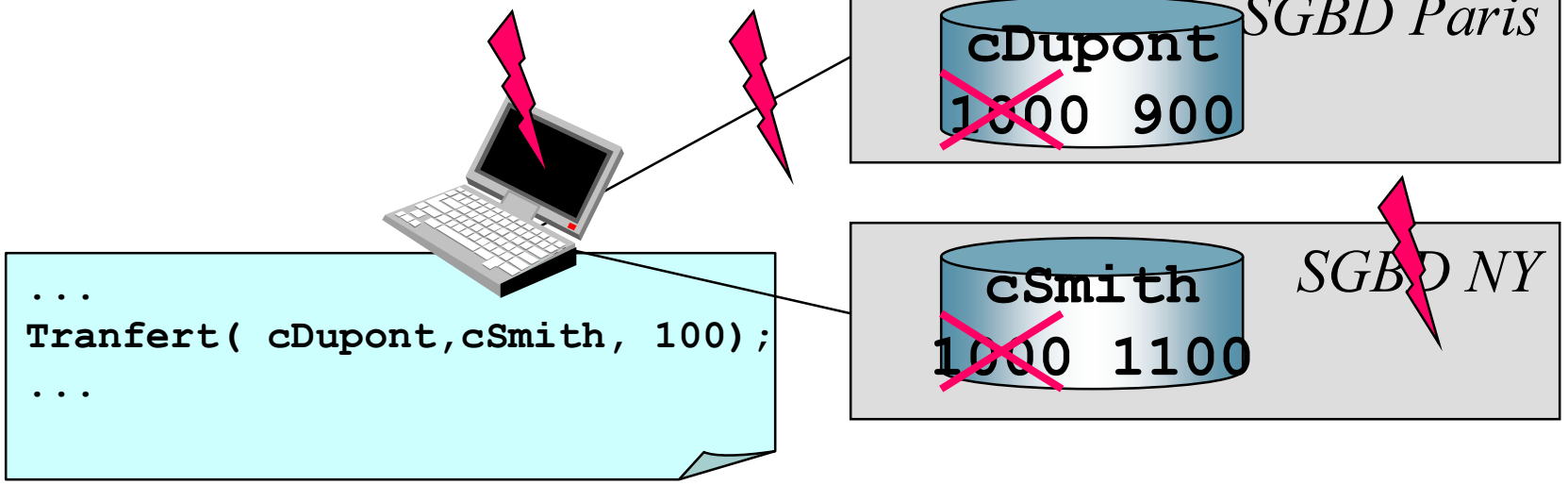
- Hot Stanby d ’Oracle



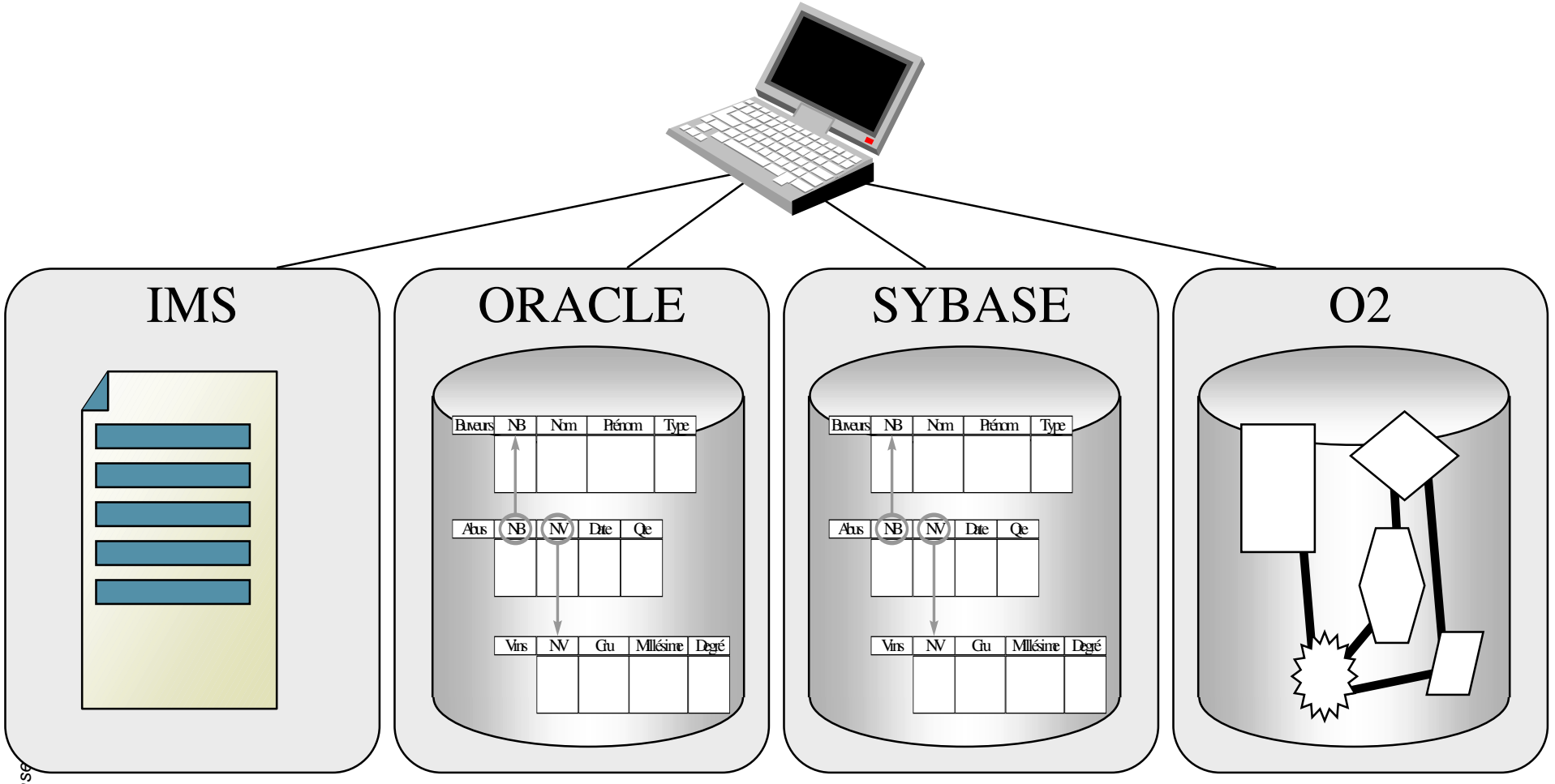
Les Systèmes Transactionnels

- Transaction
  - propriétés ACID
- Exemple Débit - Crédit
- Moniteurs Transactionnels

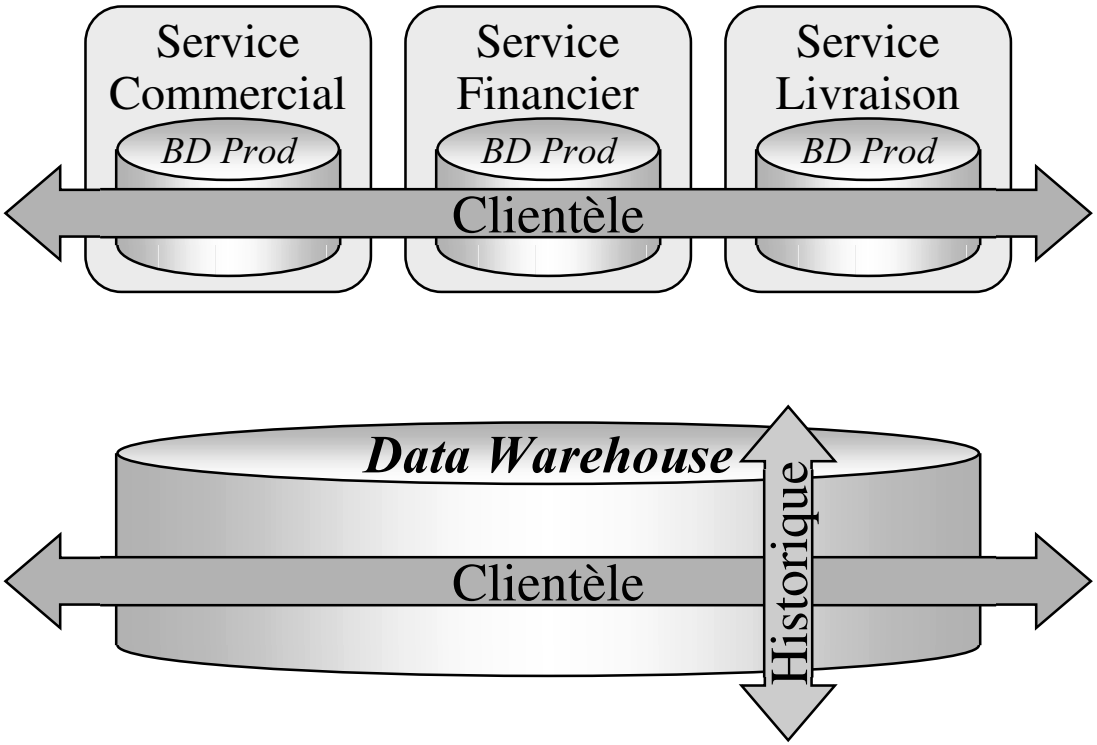
```
Tranfert(cptA,cptB,V) {
  A := READ(cptA);
  A := A-V;
  WRITE(cptB,B);
  B := READ(cptB);
  B := B+V;
  WRITE(cptB,B);
}
```



les SGBDs Hétérogènes



les Entrepôts de Données



Benchmark Bases de Données

- Banc de Performances
  - Mesurer les performances d'un système (matériel / logiciel) sous une charge de travail caractérisant une application modèle.
- Intérêt:
  - Comparer avec d'acheter
  - Dimensionner son système en fonction de ses besoins
  - MAIS ATTENTION à l'écart entre Application Réelle et Application «Modèle»

Résultats attendus d ’un Benchmark

Indicateurs

- Performance sur un système donnée
  - Nb de Transactions réalisées par seconde
- Coût du système complet (matériel+logiciel+maintenance)
  - Prix par 1 tps

Exemple : le TPC-D

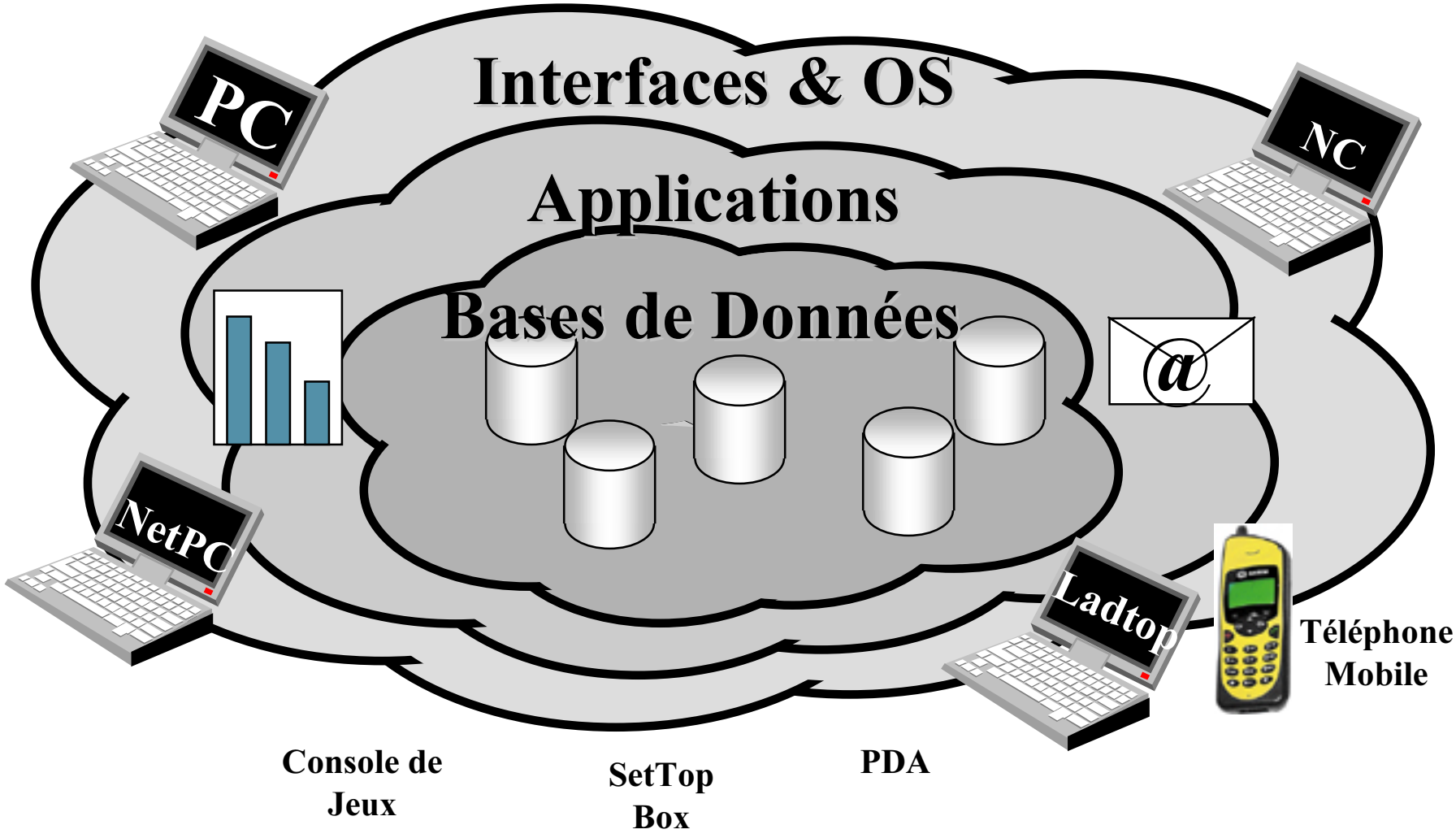
Machine Sequent NUMA Q2000 + Base de 300 Go

- + ORACLE 8
  - Puissance : 3232,3 TPC (soit 19700 F / tps)
- + INFORMIX XPS
  - Puissance : 2667,7 TPC (soit 20800 F / tps)

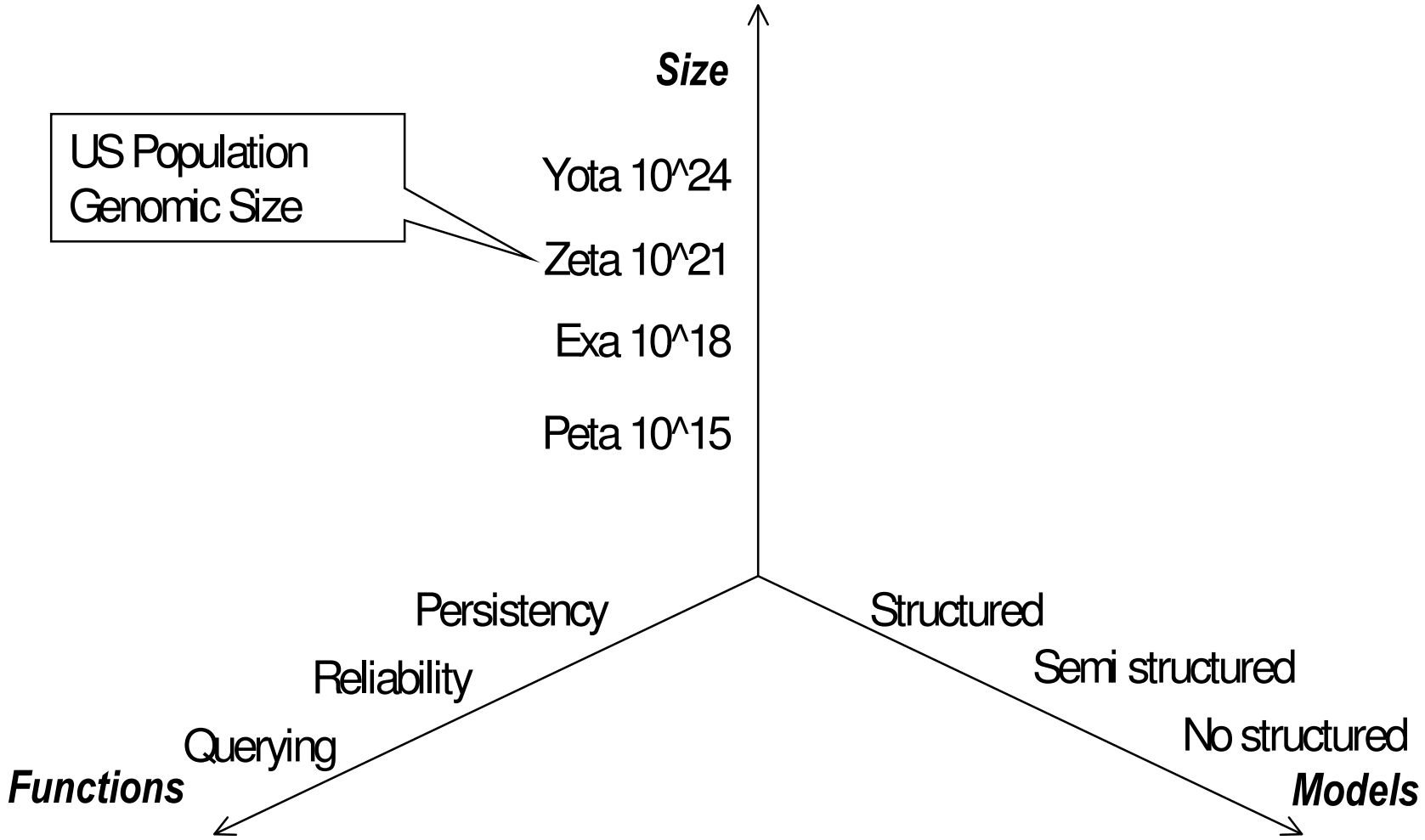
TPC (http://www.tpc.org)
Transaction Processing-performance Council

- corporation de 44 entreprises *(San José, 1988)*
  - But: Définir des benchmarks
    - pour des SGBDs
    - pour des Moniteurs Transactionnels
  - 5 benchmarks BD
    - TPC/A et TPC/B — Transactionnel (OLTP)
    - TPC/C et Draft TPC/E — New Order
    - TPC/D — Décisionnel (OLAP)
  - d 'autres benchmarks
    - TPC-W — Web et Commerce Electronique

le Système d'Information d'une entreprise

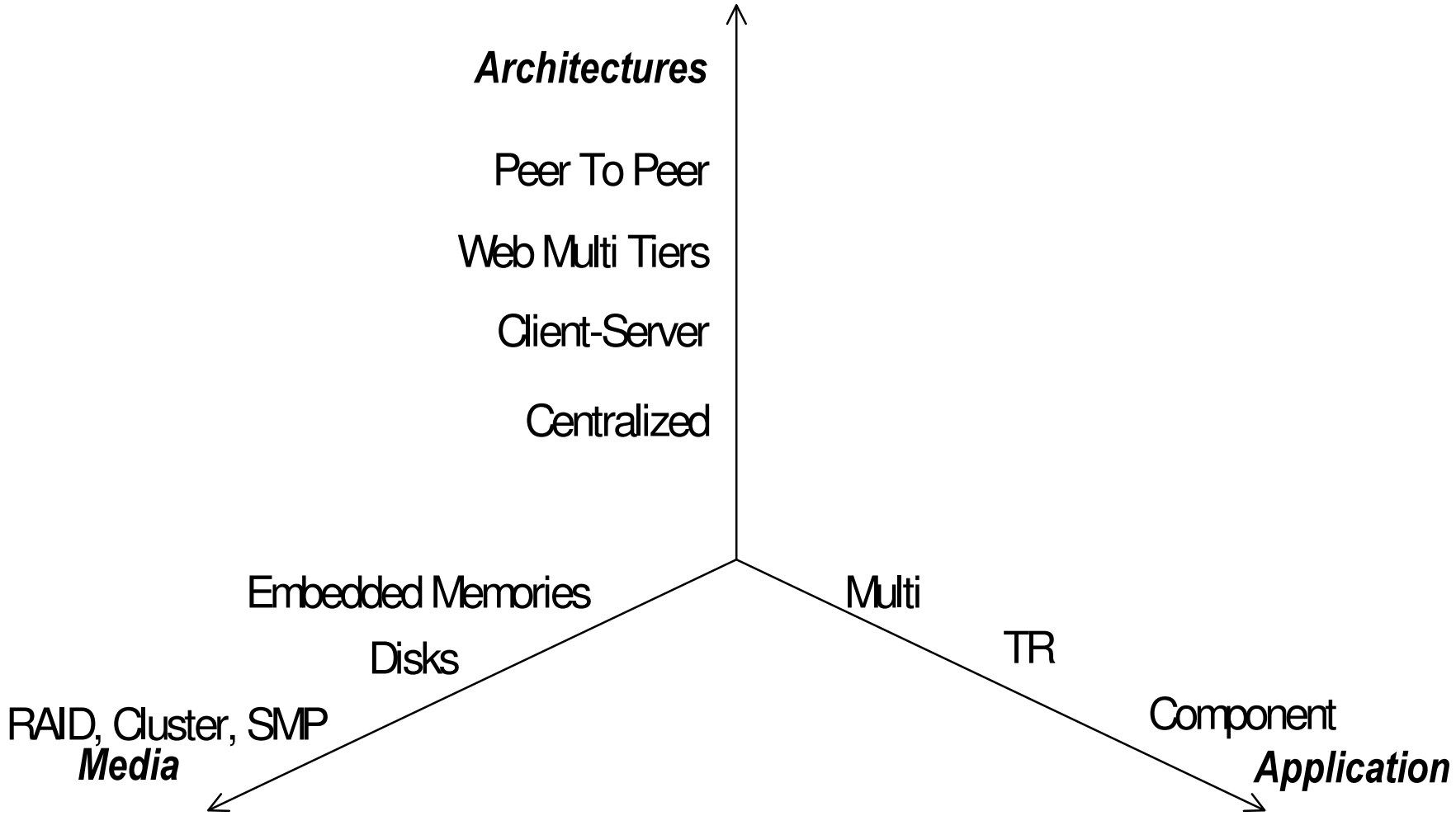


Le Futur (i)



D'après C. Collet, 2002

Le Futur (ii)



D'après C. Collet, 2002