Introduction aux concepts des bases de données

m2iformation.fr

UTOPIOS®
TRANSFORMATION DIGITALE INTELLIGENTE

Introduction aux concepts des bases de données

Qu'est-ce qu'une donnée ?

En informatique, les **données** (**data** en anglais) sont des **représentations d'informations**
Nous pouvons les **stocker**, les **traiter** et les **manipuler**

Elles peuvent prendre des formes ou des **types** variés.
Exemple: données binaires, données numériques, données textuelles...

Les données (data)

Je m'appelle Jean Dupont, je suis garagiste et j'ai 25ans

Je m'appelle Marie Dupuis, je suis chauffagiste et j'ai 33ans

Je m'appelle Eric François, je suis contrôleur et j'ai 51ans

Les données (data)

Je m'appelle Jean Dupont, je suis garagiste et j'ai 25ans

Je m'appelle Marie Dupuis, je suis chauffagiste et j'ai 33ans

Je m'appelle Eric François, je suis contrôleur et j'ai 51ans

Données structurées, semi-structurées, non structurées

Les données structurées :

Données organisées dans des formats fixes comme des **tables** (lignes et colonnes). Leur **format** est prédéfini et sont **faciles à manipuler**.

Prénom	Nom	Métier	Age
Jean	Dupont	garagiste	25
Marie	Dupuis	chauffagiste	33
Eric	François	controleur	51

Le format CSV

Le fichier représente un **tableau** dont les données sont séparées par des **virgules**, des points-virgules ou des tabulations
La première ligne du fichier donne le nom des **colonnes**

```
year,position,artist,song,indicativerevenue,us,uk,de,fr,ca,au
"2000","1","Faith Hill","Breathe","24030.051","2","33","-","-","-",
"2000","2","Santana & The Product G","Maria Maria","23320.084","1",
"2000","3","Joe Thomas","I Wanna Know","21516.777","4","-","-","61"
"2000","4","Aaliyah","Try Again","21099.824","1","5","5","26","-","
"2000","5","Toni Braxton","He Wasn't Man Enough","21023.066","2","5
"2000","6","Rob Thomas & Santana","Smooth","20735.418","1","3","21"
"2000","7","Vertical Horizon","Everything You Want","20402.965","1"
"2000","8","Destiny's Child","Say My Name","19489.657","1","3","14"
"2000","9","Lonestar","Amazed","19138.169","1","21","91","-","-","1
"2000","10","Matchbox Twenty","Bent","18997.978","1","-","-","-","1
"2000","11","Madonna","Music","18983.471","1","1","2","8","1","1"
"2000","12","Sisqo","Thong Song","18403.832","3","3","15","15","-",
"2000","13","Three Doors Down","Kryptonite","18341.509","3","-","85
"2000","14","Destiny's Child","Jumpin' Jumpin'","18020.444","3","5"
"2000","15","Creed","Higher","17082.223","7","47","-","-","-","-"
```

source : Chart2000, version 0.3.0067

Le tableur

Un programme qui manipule des **feuilles de calcul**, et qui représente également les données en **tableaux**.

VisiCalc,le premier tableur (1979)

Les données semi-structurées :

Données **partiellement** organisées avec une structure **flexible**. Elles ne rentrent pas dans des tables classiques, mais possèdent des balises ou des paires clé/valeur qui facilitent l'analyse (JSON, YAML).

Exemple d'un fichier JSON:

```
{
  "id": 1,
  "nom": "Alice",
  "email":"alice@example.com",
  "telephone": "06 12 34 56 78"
}
```

Les données non structurées :

Abondantes dans le **Big Data**, elles sont faciles à accumuler, riches en information mais complexe à traiter.

Elles ne peuvent **pas** être correctement représentées sous la forme d'un tableau

Exemples : e-mails, photos, fichiers audio...

Éric François <eric.francois@mail... 09:30 (il y a 6 minutes)
À Marie Dupuis

Joyeux anniversaire Marie ! J'espère que tu fêtes ça dignement, 33 ans c'est pas rien ! Comment tu vas ?

Ça fait un bail, j'espère te recroiser bientôt. T'as des nouvelles de Jean ? Je me demandais si il était toujours garagiste.

Si ça t'intéresse, je connais une pâtisserie qui fait des fraisiers super bons, celle qui est rue Fragaria. Pas d'anniversaire sans gâteau !

À bientôt,

Éric

Nom	Prénom	Métier	Âge	Va peut-être s'acheter un fraisier
François	Éric	?	33	?
Dupuis	Marie	?	?	oui ?
?	Jean	Garagiste ?	?	?

Leur **traitement** est plus complexe : data science, intelligence artificielle, machine learning...

Les bases de données

Qu'est-ce qu'une base de données ?

Une base de données (BDD), ou database (DB), est **un ensemble organisé de données stockées** électroniquement dans un système informatique

L'histoire des bases de données remonte aux **années 1960**, ou l'on commence à pouvoir stocker **de grandes quantités de données**. Le besoin de gérer efficacement ces **grands volumes** de données a conduit à la création des BDD.

Différents types de BDD

Les BDD peuvent s'organiser de manières différentes, appelées **modèles de base de données** qui sont adaptées à **différents types de données**, et répondant à des exigences différentes : orientée texte, hiérarchique, réseau, relationnelle, orientée objet...

De nos jours, les deux modèles dominants sont :

Modèle relationnel (SQL)

Modèle non-relationnel (NoSQL)

Les bases de données relationnelles

Développées dès le début des années 70 par les travaux d'Edgar F. Codd, **les bases de données relationnelles** (ou relational databases) sont l'un des modèles de base de données les plus utilisés.

Les BDDR sont basées sur la théorie des relations entre données et utilisent des **tables** composées de **lignes** et de **colonnes** pour les organiser.
Cela signifie qu'on va essayer de regrouper nos données à travers des caractéristiques ou des étiquettes **communes**.

- Avantages :
  - **Fiabilité** : les transactions sont sûres et cohérentes (**ACID**).
  - **Structure claire** : schéma fixe avec tables, relations, clés primaires/étrangères.
  - **Langage standardisé** : requêtes avec SQL, langage universel et puissant.
  - **Intégrité des données** : grâce aux contraintes et aux types définis.
- Inconvénients :
  - **Rigidité du schéma** : difficile à modifier si les besoins évoluent.
  - **Moins adapté au big data** : difficulté à gérer des volumes massifs ou très variés.
  - **Montée en charge verticale** : plus difficile à distribuer horizontalement.

Les bases de données non relationnelles

Face à l'explosion du volume de données, à la diversité des formats (texte, images, logs, JSON, etc.) et aux besoins de **scalabilité**, sont apparues les **bases de données non relationnelles**, souvent appelées NoSQL.

Contrairement aux bases relationnelles, elles n'utilisent **pas de schéma tabulaire fixe**. Elles permettent de stocker et manipuler des données structurées, semi-structurées ou non structurées **de manière plus souple et adaptée** à certains contextes.

- Avantages :
  - **Scalabilité horizontale** : facile à répartir sur plusieurs serveurs.
  - **Souplesse des données** : pas de schéma fixe, formats variés (JSON, clé-valeur…).
  - **Performance** : très efficaces pour des lectures/écritures à grande échelle.
  - **Adapté au web moderne** : temps réel, IoT, applications distribuées.
- Inconvénients :
  - **Pas toujours ACID** : fiabilité parfois moindre (préférence pour BASE).
  - **Manque de standardisation** : chaque système NoSQL a ses propres requêtes et outils.
  - **Complexité des relations** : gestion des jointures ou cohérence plus difficile.

UTOPIOS®
TRANSFORMATION DIGITALE INTELLIGENTE

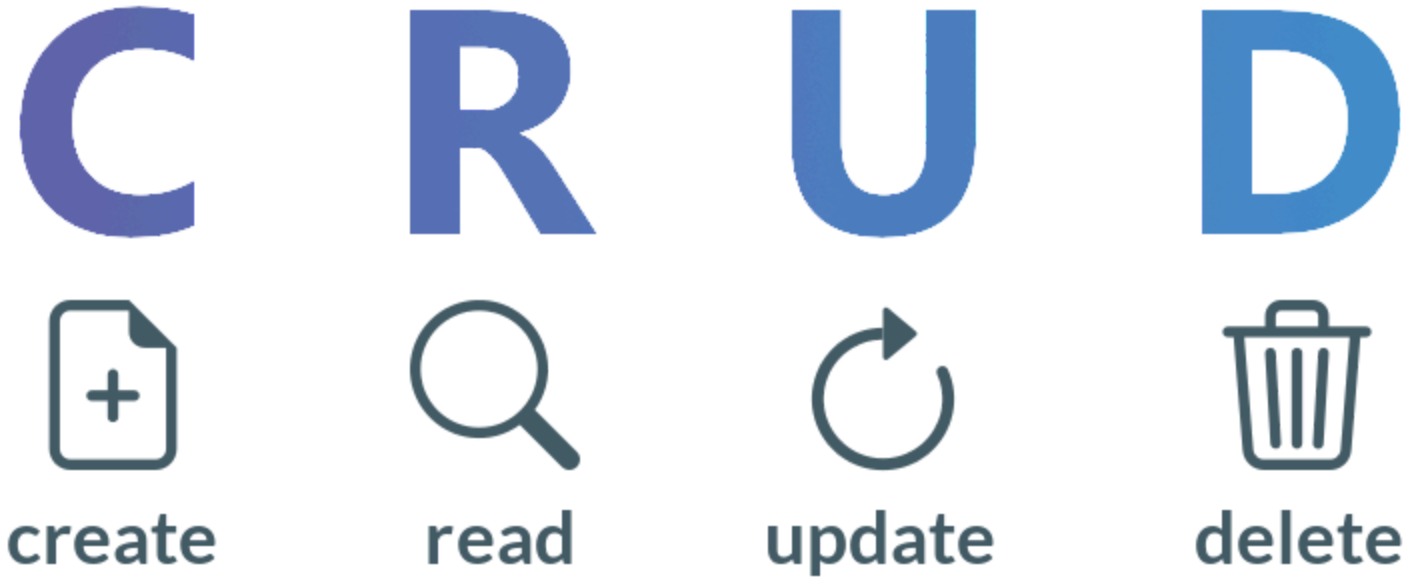
Les Systèmes de Gestion de Bases de Données (SGBD)

Interagir avec la base : le SGBD

Le **système de gestion de base de données** (SGBD), ou database management system (DBMS), est un **logiciel** qui nous facilitent le travail pour le stockage, la manipulation et la gestion de données.

Elles offrent un panel d'outils étendu pour nous permettre une maintenance aisée de grande quantité de données. Le SGBD sert d'**intermédiaire** entre l'utilisateur/l'application et les données.

Il permet de réaliser les quatre opérations de base du **CRUD** : créer, lire, mettre à jour et supprimer



ainsi que des actions **plus avancées** : les fonctions d'agrégation, les jointures, les requêtes complexes, etc.

- **Création et définition de structure**: Permet de définir tables, colonnes, clé primaires, contraintes et relations.
- **Stockage et gestion de données**: Permet de stocker de façon organisée de grandes quantités de données, mais également de gérer leurs opérations d'insertion, de mise à jour et de suppression.
- **Langage de requête**: Ils fournissent un langage pour interroger et manipuler les données de façon compréhensible et fluide.
- **Gestion des transactions** (ACID/BASE)

- **Sécurité et autorisations**: Ils permettent de définir des niveaux d'autorisation, de contrôler la confidentialité des données et de fournir des droits d'accès aux utilisateurs.
- **Sauvegarde et récupération**: Ils offrent des fonctionnalités de sauvegarde et de restauration afin d'éviter la perte de données essentielles en cas de panne système.
- **Optimisation des performance**: Ils utilisent des techniques d'optimisation telles que l'indexation, la mise en cache, la parallélisation des requêtes etc...

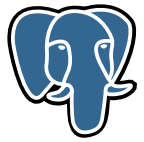
Exemples de SGBD **relationnels** (SGBDR) populaires:

- MySQL/MariaDB

- Oracle Database

- IBM DB2

- PostgreSQL



- Microsoft SQL Server

- SQLite

Exemples de SGBD **non-relationnels** (SGBDNR) populaires :

- MongoDB

- Redis

- CassandraDB

- DynamoDB

UTOPIOS®
TRANSFORMATION DIGITALE INTELLIGENTE

Le principe ACID

ACID

ACID est un acronyme qui décrit les 4 propriétés garanties par les SGBD pour assurer la fiabilité des transactions.

Une **transaction** est un ensemble d'opérations exécutées comme **une seule unité logique de travail**.
Elle représente une **action complète** sur les données (ex : virement bancaire, ajout d'une commande…), qui doit être :

- soit **entièrement exécutée**
- soit **entièrement annulée**

1. **Atomicité (Atomicity):**
   Une transaction est une unité de travail indivisible. Soit toutes les opérations de la transaction sont effectuées avec succès, soit aucune ne l'est. Si une partie de la transaction échoue, toutes les modifications effectuées par la transaction sont annulées (rollback).

2. **Cohérence (Consistency):**
   Une transaction doit transformer la base de données d'un état valide à un autre état valide. Par exemple, si une règle stipule que le solde d'un compte ne peut pas être négatif, cette règle doit être respecté avant et après la transaction.

3. **Isolation (Isolation):**
   Les transactions doivent être isolées les unes des autres. Les opérations d'une transaction ne doivent pas interférer avec celles d'une autre. Cela garantit que les transactions concurrentes ne voient pas les modifications intermédiaires des autres.

4. **Durabilité (Durability):**
   Une fois qu'une transaction est validée (commit), ses modifications doivent être permanentes, même en cas de panne. Les données doivent être enregistrées sur un support non volatile.

Le principe BASE

BASE

BASE est un acronyme qui décrit les propriétés des systèmes de bases de données NoSQL, qui sont souvent plus flexibles et tolérants aux pannes que les systèmes ACID. Les trois propriétés de BASE sont:

1. **Basically Available (Disponibilité de base):**
   Le système garantit une disponibilité de base, ce qui signifie qu'il répondra toujours à une requête, même si certaines parties du système sont en panne. Cela peut impliquer que le système retourne des données qui ne sont pas à jour ou complètes.

2. **Soft state (état souple):**
   Le système peut être dans un état intermédiaire ou incohérent pendant une courte période. Les données peuvent ne pas être immédiatement cohérentes, mais elles finiront par converger vers un état cohérent.

3. **Eventual consistency (Cohérence éventuelle):**
   Le système garantit que toutes les copies des données finiront par être cohérentes, mais il peut y avoir un délai avant que cela ne se produise. Cela signifie que les lectures peuvent ne pas refléter les écritures les plus récentes immédiatement.

Comparaison ACID vs BASE

- ACID est souvent utilisé dans les SGBDR où la cohérence et la fiabilité sont **cruciales**, comme dans les systèmes bancaires ou les systèmes de réservation.
- BASE est souvent utilisé dans les systèmes où la disponibilité et la tolérance aux pannes sont plus importantes que la cohérence immédiate, comme dans les systèmes de réseaux sociaux ou les systèmes de recommandation.

UTOPIOS®
TRANSFORMATION DIGITALE INTELLIGENTE

Le théorème CAP

Le Théorème CAP

Le théorème CAP, formulé par Eric Brewer, stipule que **dans un système distribué**, il est impossible de garantir **simultanément** les trois propriétés suivantes **en cas de partition réseau** :

- **C – Cohérence (Consistency)** : Toutes les requêtes reçoivent la même réponse, même sur des nœuds différents.
- **A – Disponibilité (Availability)** : Le système répond toujours à toutes les requêtes, même si certaines données ne sont pas à jour.
- **P – Tolérance au partitionnement (Partition Tolerance)** : Le système continue de fonctionner même si des parties du réseau sont coupées ou défaillantes.

Consistency + Availability : CA

Ce modèle garantit une **cohérence forte** et une **disponibilité continue**, mais ne résiste pas bien aux **partitions réseau** (pas de tolérance au partitionnement).

On retrouve ce modèle dans de nombreux **SGBD relationnels**, notamment lorsqu'ils fonctionnent sur **un seul serveur** ou en **local**.

➡ Les données sont **stockées de manière centralisée**, ce qui peut limiter la **scalabilité** et rendre le système plus vulnérable en cas de panne ou de surcharge.

Consistency + Partition Tolerance : CP

Ce modèle garantit une **cohérence forte** et une **résilience aux partitions réseau**, mais **sacrifie la disponibilité** dans certaines situations (le système peut refuser des requêtes plutôt que de risquer l'incohérence).

➡ Ce modèle nécessite une **synchronisation constante entre les nœuds**, ce qui implique souvent des **temps de réponse plus longs** et une **latence plus élevée**.

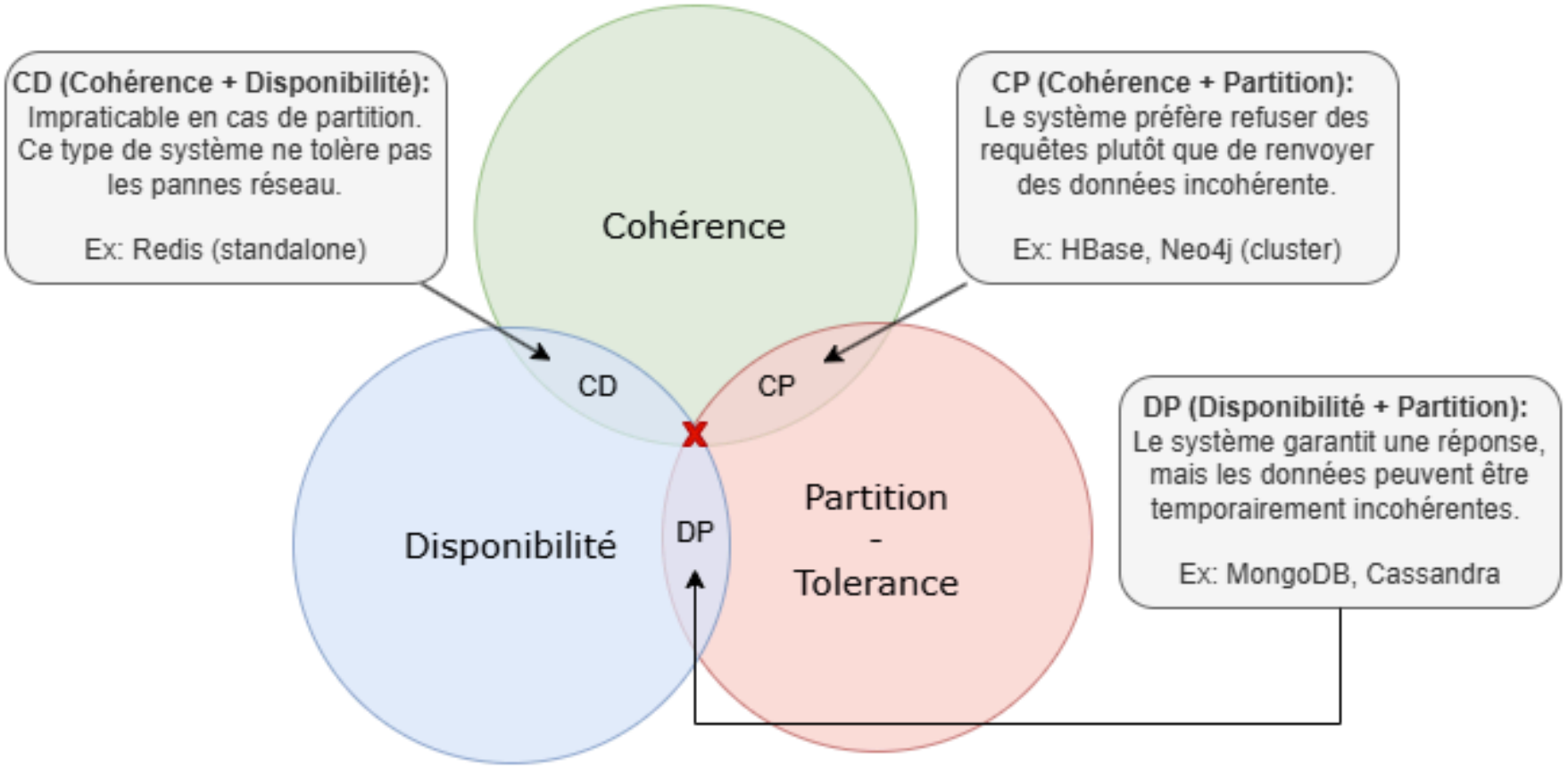
Availability + Partition Tolerance : AP

Ce modèle garantit une **disponibilité élevée** et une **résilience aux partitions réseau**, mais sacrifie la **cohérence immédiate**.

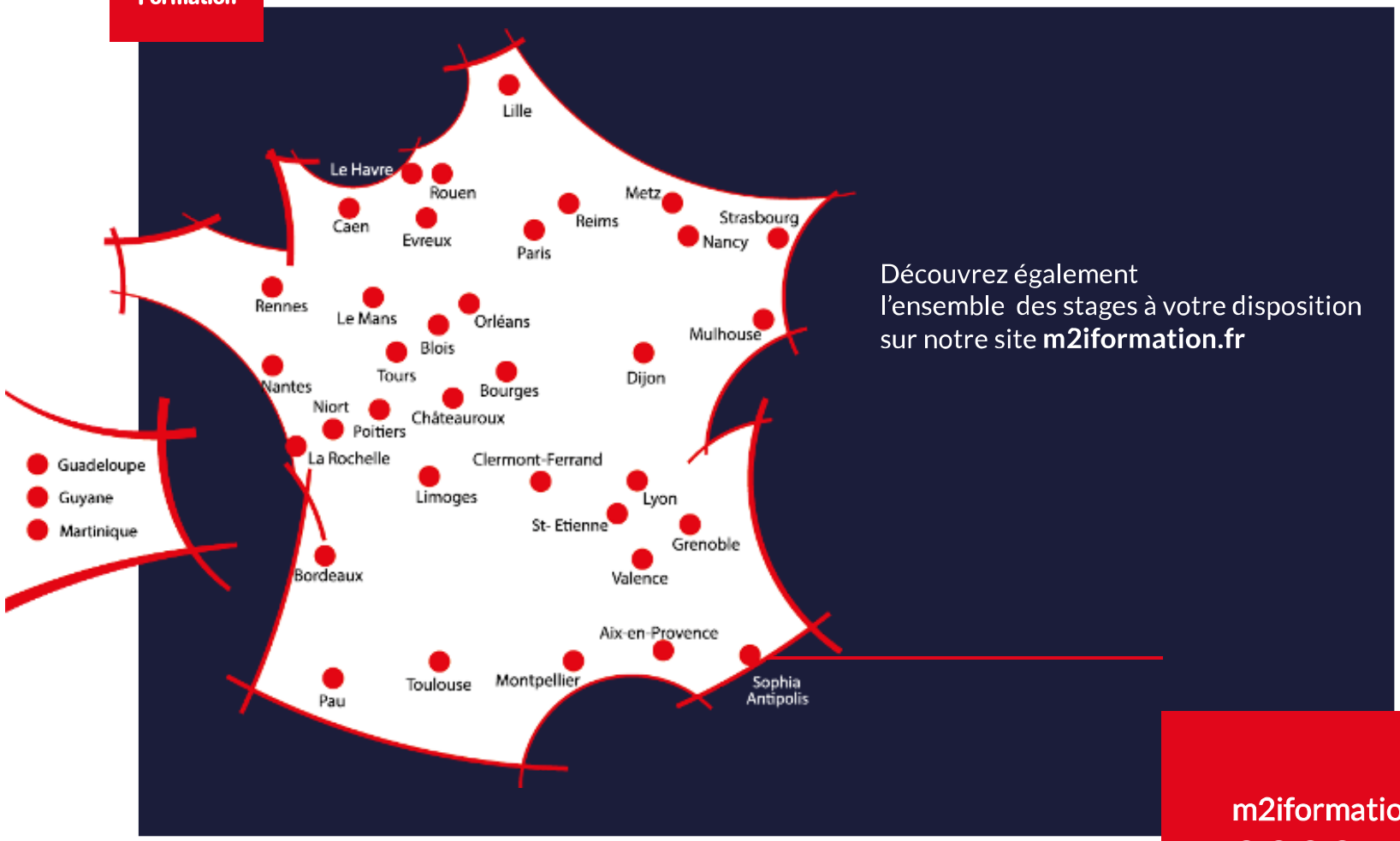
Les données sont **distribuées** sur plusieurs nœuds, et le système continue de répondre même si certains nœuds sont inaccessibles ou désynchronisés. On parle alors de **cohérence éventuelle**.

➡ Ce modèle offre des **temps de réponse très rapides**, mais peut entraîner des **divergences temporaires** entre les copies de données, ainsi que des **conflits** à résoudre.

Triangle CAP : le compromis



➡ D'où l'intérêt de **bien choisir** sa base de données en fonction de ses **besoins.**



Découvrez également
l'ensemble des stages à votre disposition
sur notre site **m2iformation.fr**

m2iformation.fr