

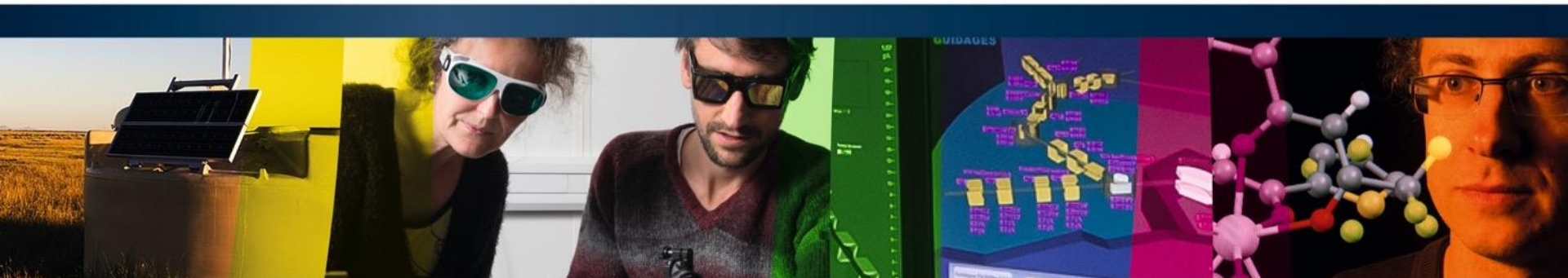


www.cnrs.fr

NoSQL

LILLE

16 au 18 mai 2017



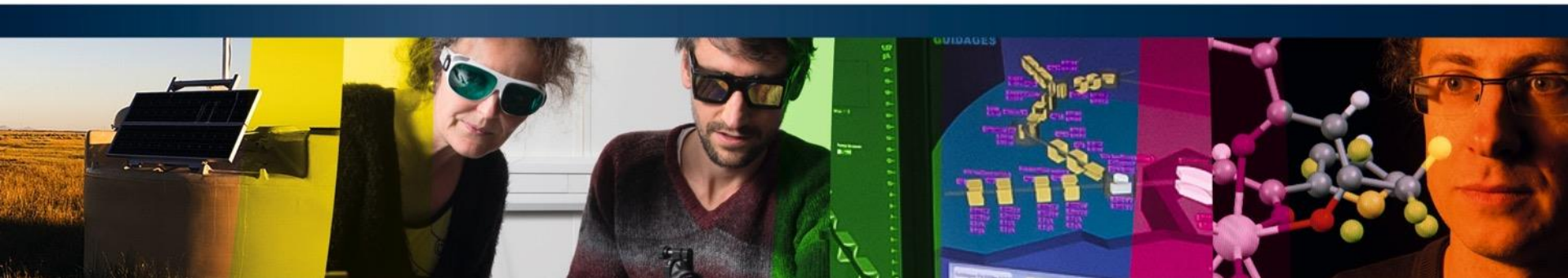
Guillaume HARRY



www.cnrs.fr

NoSQL

Introduction



Guillaume HARRY



Sommaire

- 1. Histoire des bases de données**
- 2. Le mouvement NoSQL**
- 3. Les grandes familles du NoSQL**



Histoire des bases de données



- ⦿ Qu'est-ce qu'une base de données ?
- ⦿ Le début des SGBD
- ⦿ Les SGBD Relationnels
- ⦿ Le mouvement BigData

Qu'est-ce qu'une base de données ?



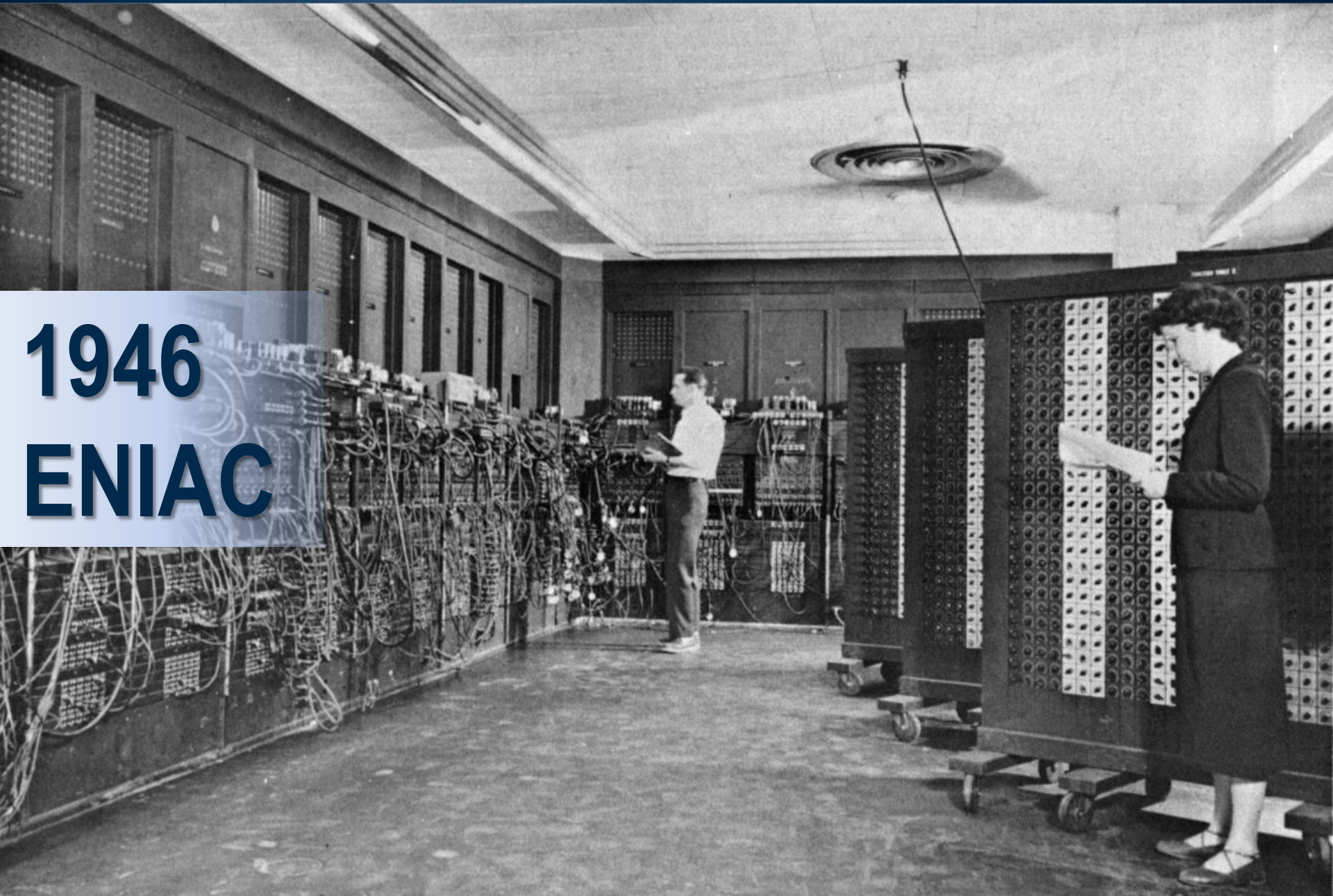
**Stocker et retrouver
les informations**

Les débuts des bases de données

Bibliothèque d'Alexandrie

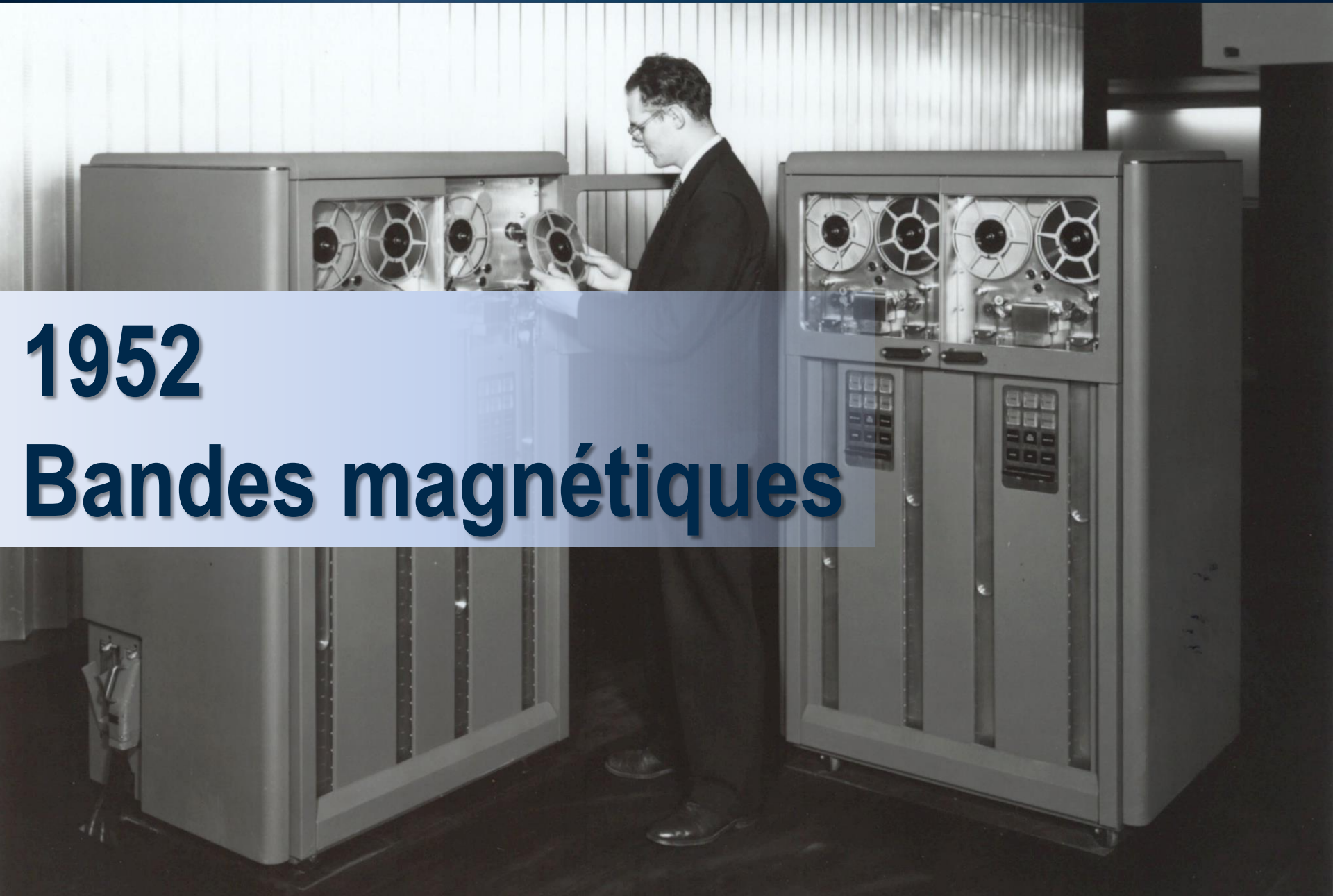


Les débuts des SGBD



1946
ENIAC

Les débuts des SGBD



1952

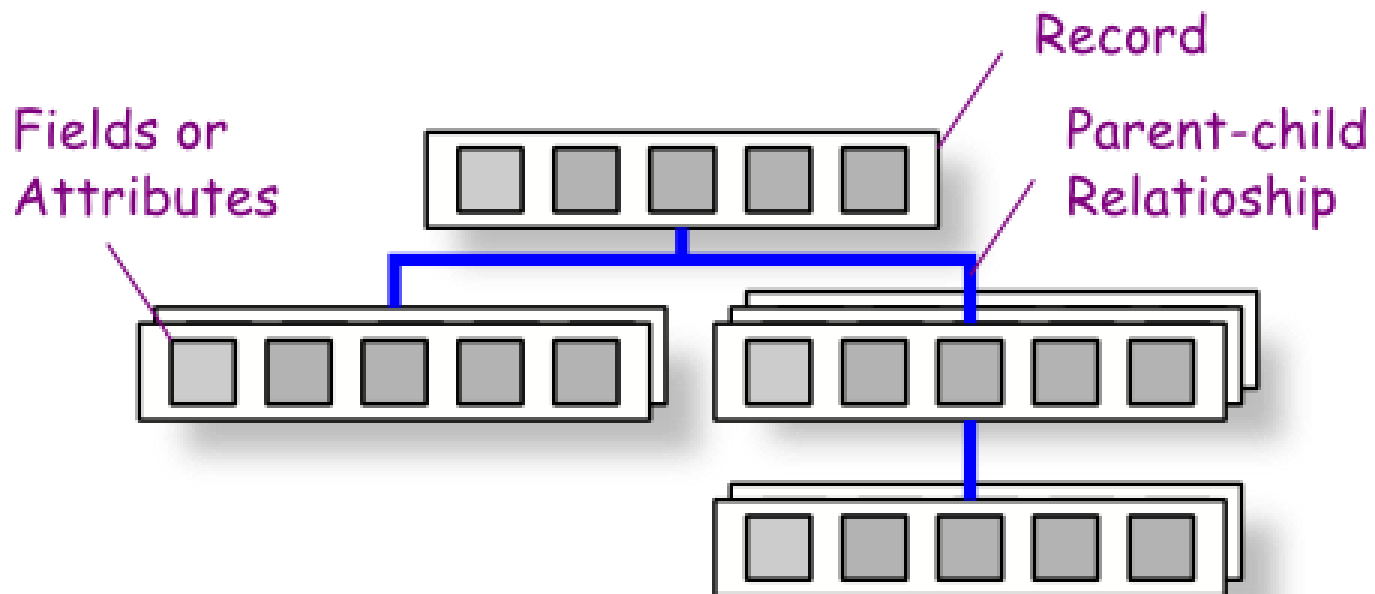
Bandes magnétiques

Les débuts des SGBD

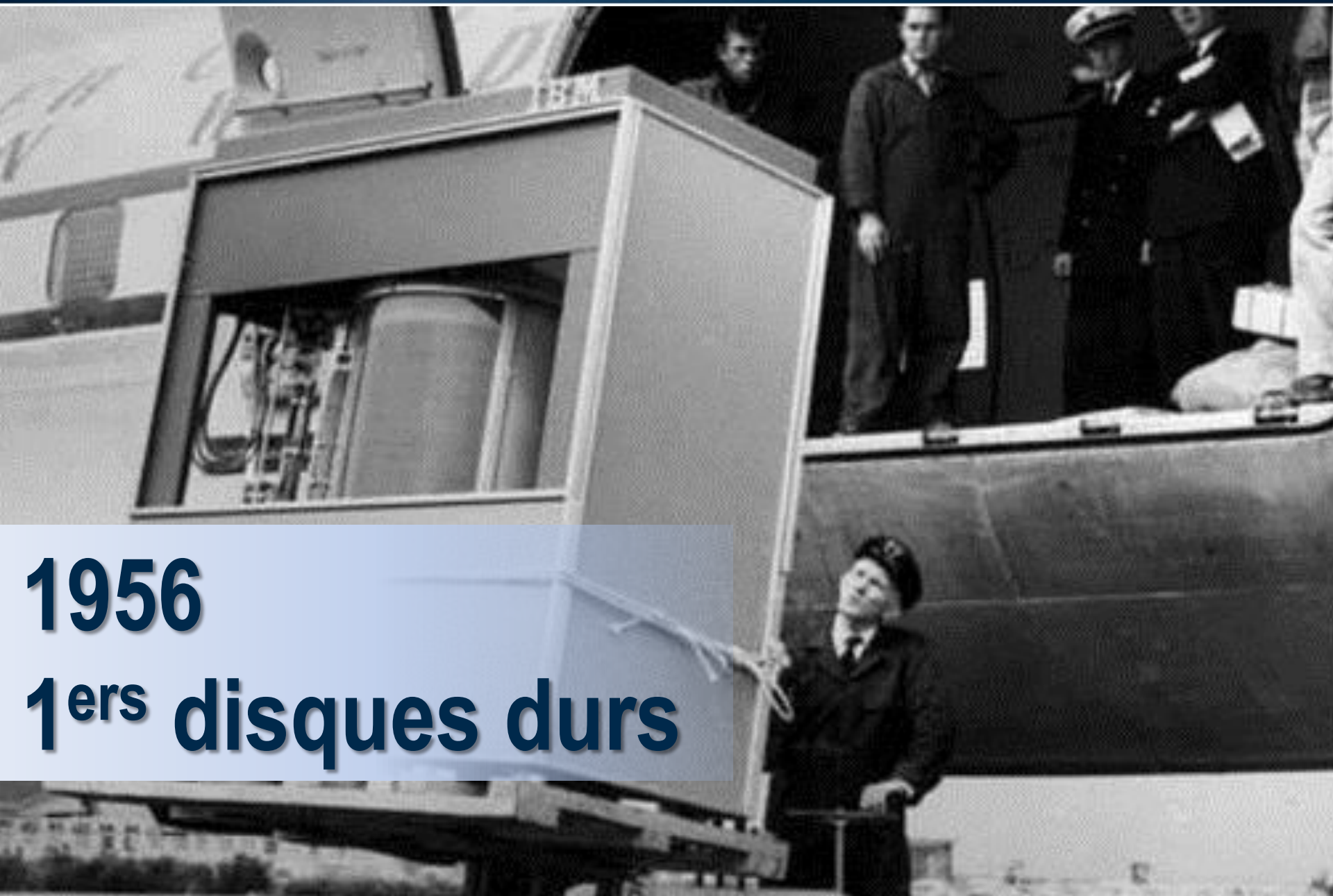
🕒 Modèle « hiérarchique »

Né dans les années 1950

Toujours utilisé



Les débuts des SGBD



1956

1^{ers} disques durs

Les débuts des SGBD



1956

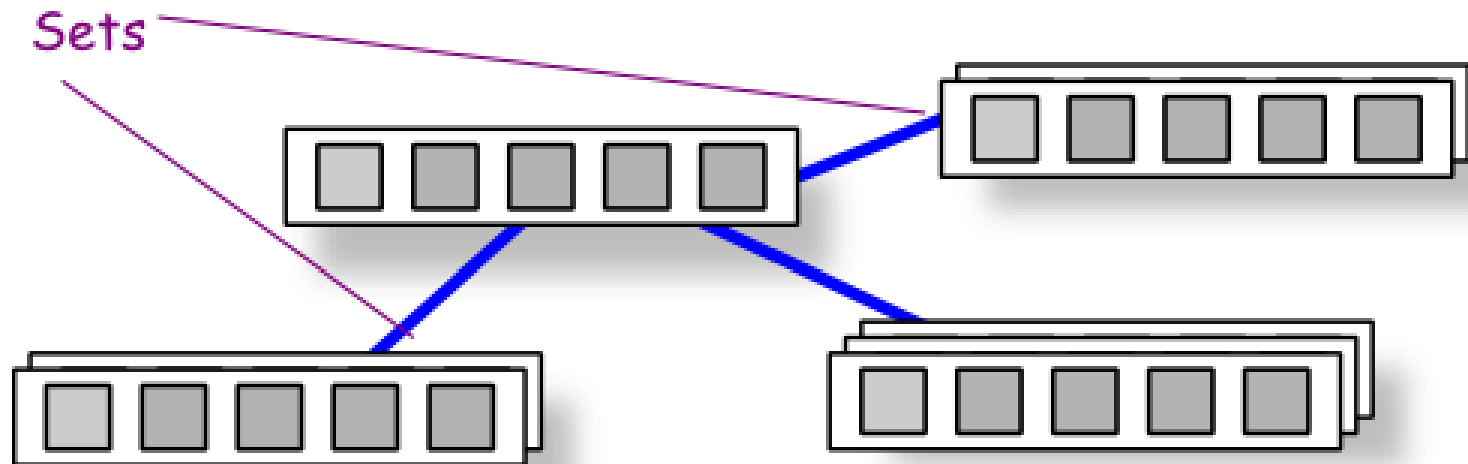
1^{ers} disques durs

Les débuts des SGBD

🕒 Modèle « réseau »

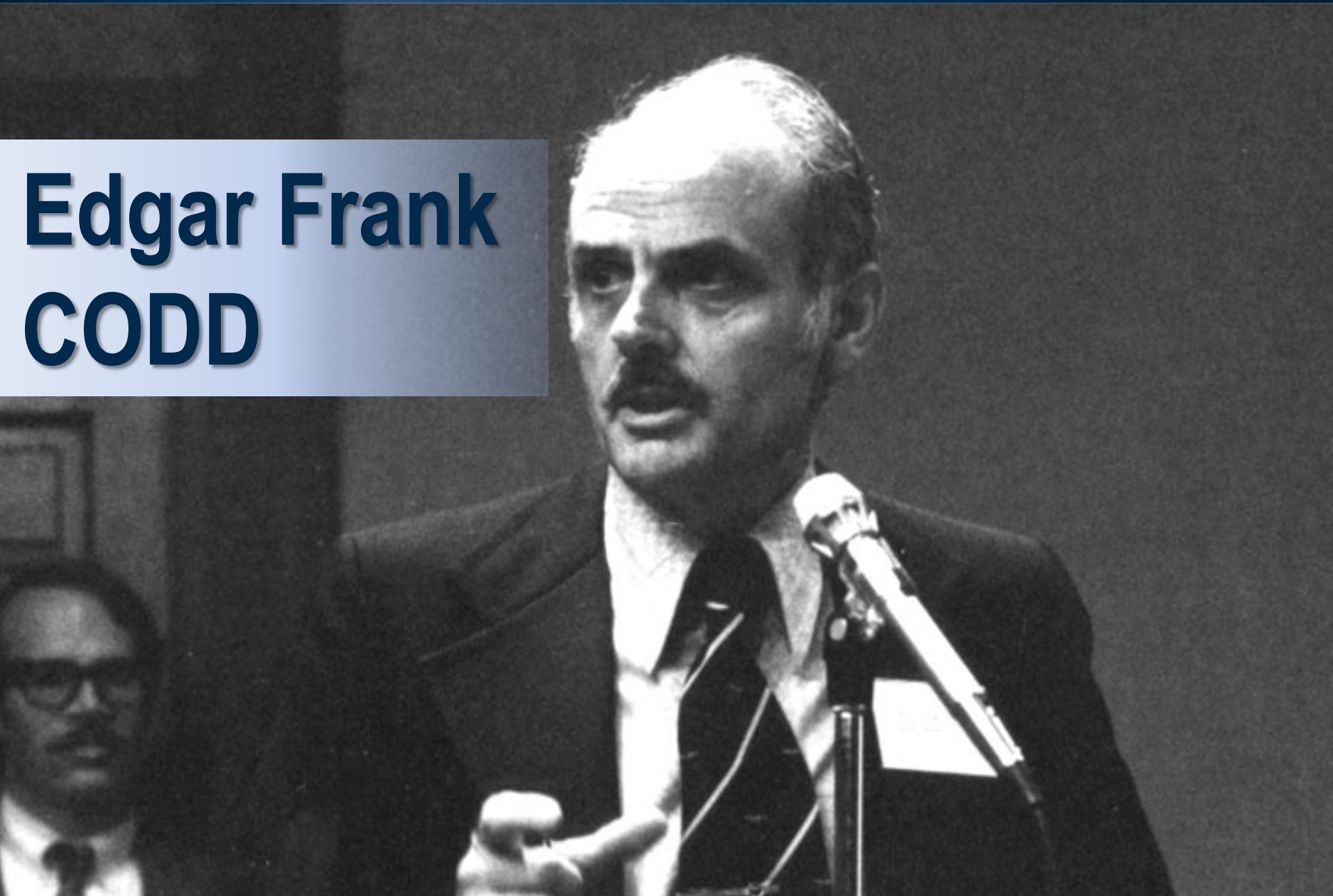
Né dans les années 1960

Langage Cobol (COmmon Business Oriented Language)



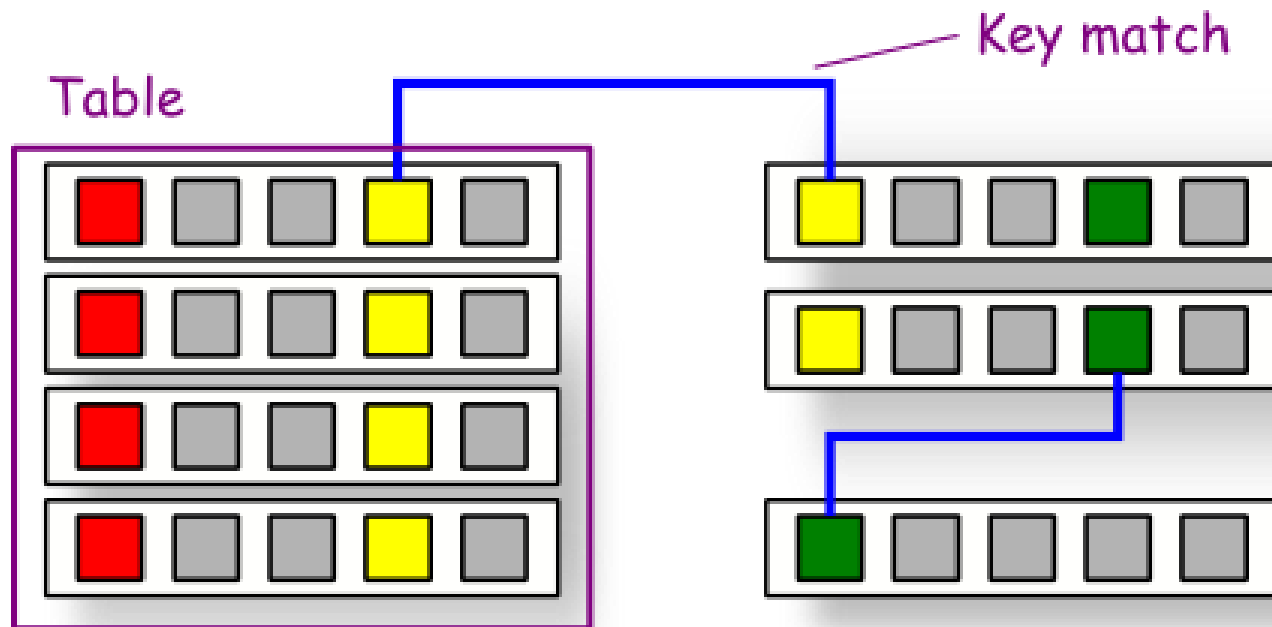
Les SGBD Relationnels

**Edgar Frank
Codd**



Les SGBD Relationnels

Le modèle relationnel



Les SGBD Relationnels

🕒 Les 12 règles de CODD

RÈGLE 1 - Règle de l'information

Toutes les informations dans une base de données relationnelle sont représentées de façon explicite au niveau logique et d'une seule manière : par des valeurs dans des tables.

RÈGLE 2 - Garantie d'accès

Toute donnée atomique d'une base de données relationnelle est accessible par le biais d'une combinaison du nom de la table ou de la vue, d'une valeur de la clé primaire et d'un nom de colonne.

RÈGLE 3 - Gestion du marqueur NULL

Les marqueurs NULL sont systématiquement pris en charge pour représenter des informations manquantes ou inapplicables, indépendamment du type, du domaine de valeur ou d'une valeur par défaut.

Les SGBD Relationnels

🕒 Les 12 règles de CODD

RÈGLE 4 - Catalogue relationnel, dynamique et accessible directement

La description de la base de données et de son contenu est représentée au niveau logique de la même manière que les données ordinaires (des tables).

RÈGLE 5 - Langage de manipulation de données complet

Au moins un des langages du SGBDR doit avoir une syntaxe complète et doit permettre la définition des données, la formation des vues, la manipulation des données, la gestion des règles d'intégrité, les autorisations et les frontières des transactions.

RÈGLE 6 - Règle de mise à jour des vues

Toutes les vues qui sont théoriquement modifiables peuvent être mises à jour par le système.

RÈGLE 7 - Insertion, suppression et modification ensemblistes

Le SGBDR retourne un ensemble d'éléments en réponse aux requêtes qui lui sont soumises. Il doit pouvoir mettre à jour un ensemble d'éléments en exécutant une seule requête.

Les SGBD Relationnels

🕒 Les 12 règles de CODD

RÈGLE 8 - Indépendance physique des données

Les applications et les programmes terminaux sont logiquement inaffectés lorsque les méthodes d'accès physiques ou les structures de stockage sont modifiées.

RÈGLE 9 - Indépendance logique des données

Les applications et les programmes terminaux sont logiquement inaffectés, quand des changements de tous ordres, préservant les informations et qui ne leur portent théoriquement aucune atteinte, sont apportés aux tables de base (restructuration).

RÈGLE 10 - Indépendance vis-à-vis de l'intégrité

Les contraintes d'intégrité spécifiques à une base de données relationnelle sont définies à l'aide du langage relationnel et leur définition doit être stockée dans le catalogue et non dans des programmes d'application.

Les SGBD Relationnels

🕒 Les 12 règles de Codd

RÈGLE 11 - Indépendance de distribution

Le langage relationnel doit permettre aux programmes d'application et aux requêtes de demeurer identiques sur le plan logique lorsque des données, quelles qu'elles soient, sont physiquement réparties ou centralisées.

RÈGLE 12 - Non subversion

Il ne doit pas être possible de transgresser les règles d'intégrité et les contraintes définies par le langage relationnel du SGBDR en utilisant un langage de plus bas niveau (gérant une seule ligne à la fois).

Les SGBD Relationnels

- ⦿ La règle qui les gouverne toutes

**L'intégralité des fonctions du SGBDR
doit être accessible par le modèle relationnel**



Les SGBD Relationnels

Gestion des transactions et des accès concurrents

⦿ Atomicité

Une série d'opérations (=« transaction ») est exécutée soit en totalité soit pas du tout.

⦿ Cohérence

Toute modification fait évoluer la base d'un état cohérent à un autre état cohérent, en respectant l'intégralité des contraintes d'intégrité, y compris les contraintes référentielles.

⦿ Isolation

Aucune transaction en cours n'est affectée par une quelconque transaction non encore validée, même si certaines des opérations qui la composent sont déjà validées séparément.

⦿ Durabilité (rémanence)

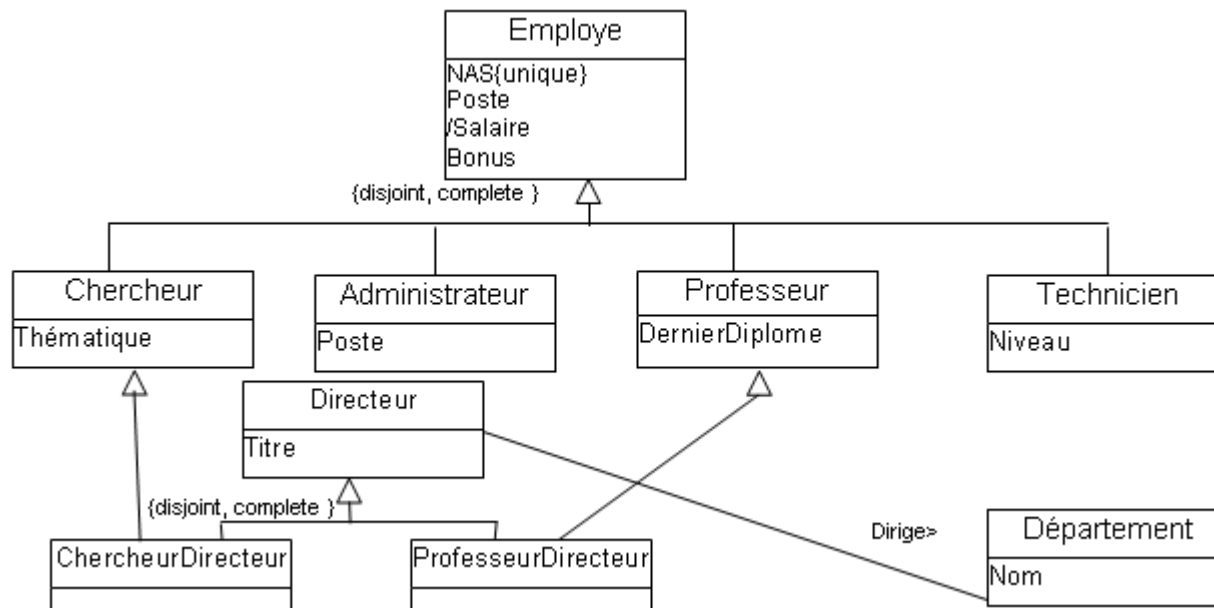
Les résultats d'une transaction une fois validée sont permanents, y compris en cas de destruction de parties du matériel non affectées au stockage des données de la base.

Les SGBD relationnels orientés objets

🕒 Modèle « orienté objet »

Né dans les années 1985

Persistance des objets définis dans les langages objets



🕒 Remplacés par les ORM (Object Relationship Management)

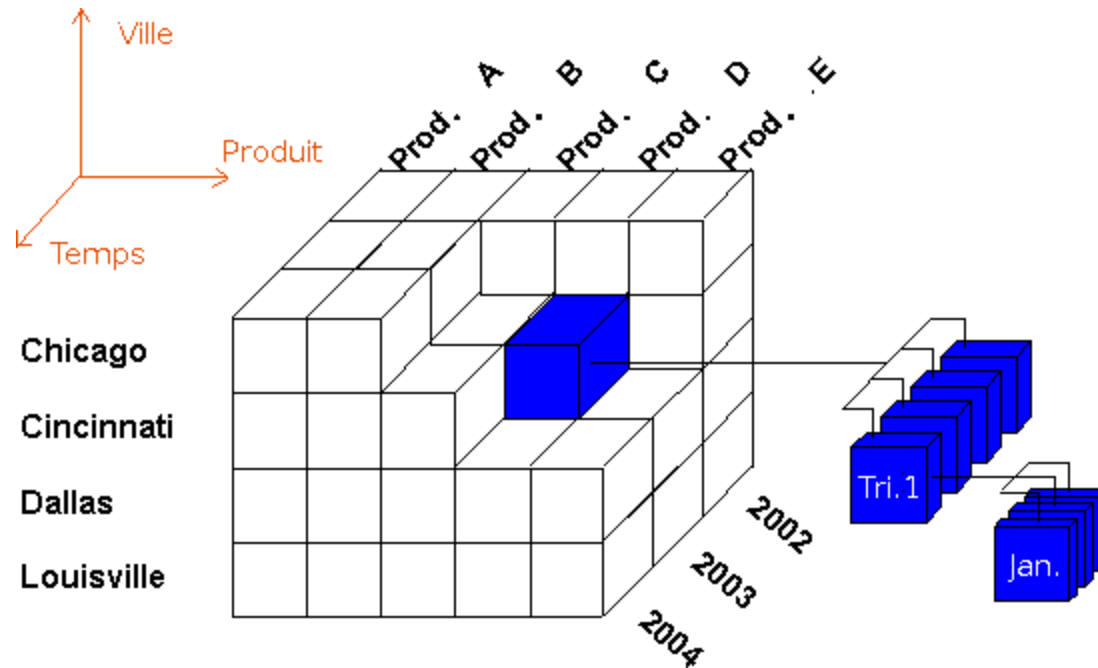
A partir de 1990

Généralisation des disques durs



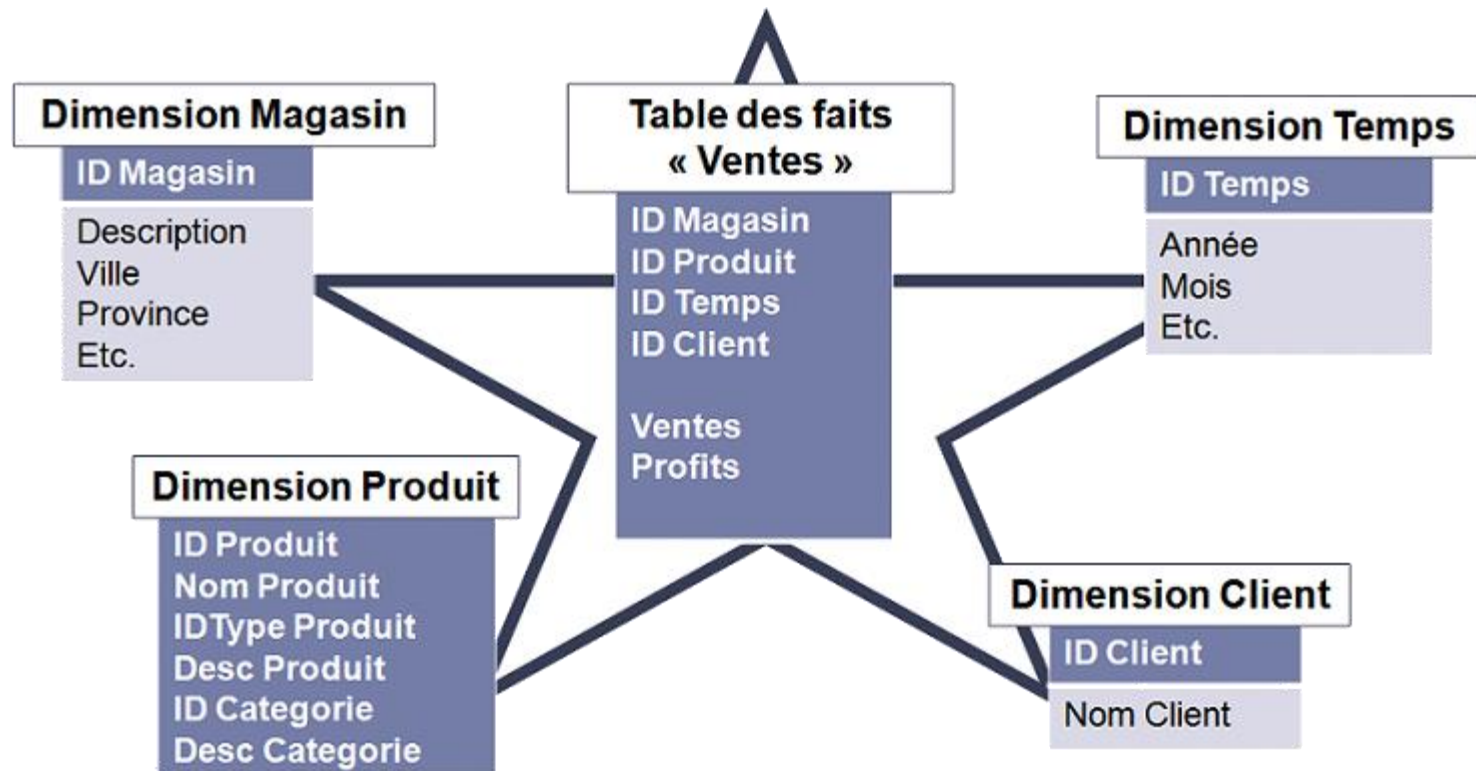
Les SGBD multi-dimensionnels

Les HyperCubes



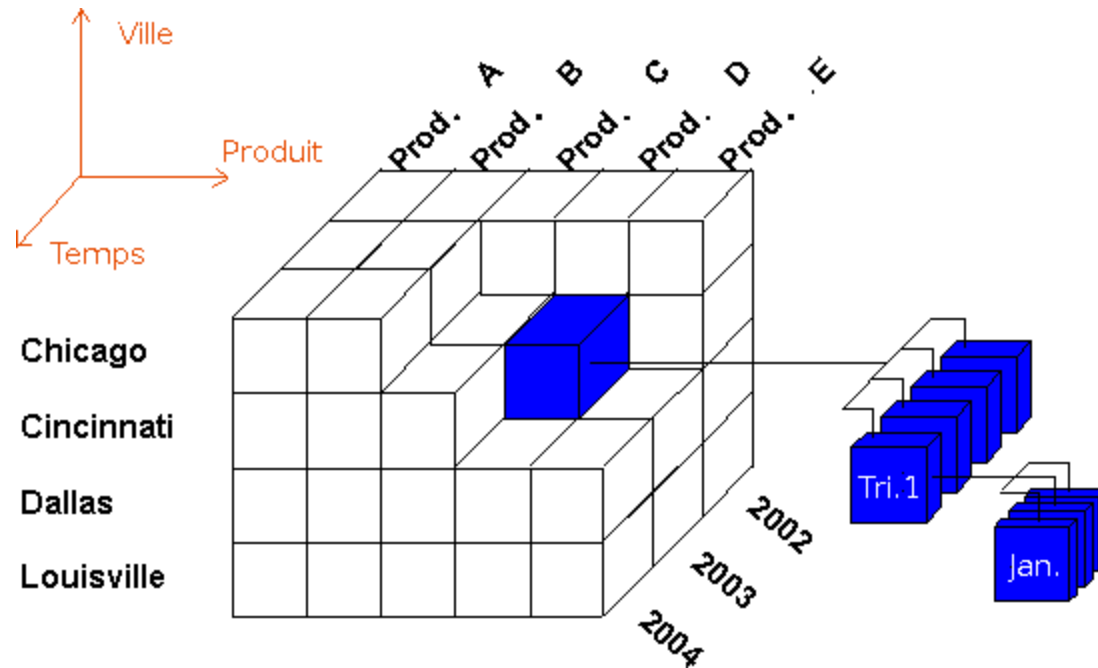
Les SGBD multi-dimensionnels

Modèle en étoile



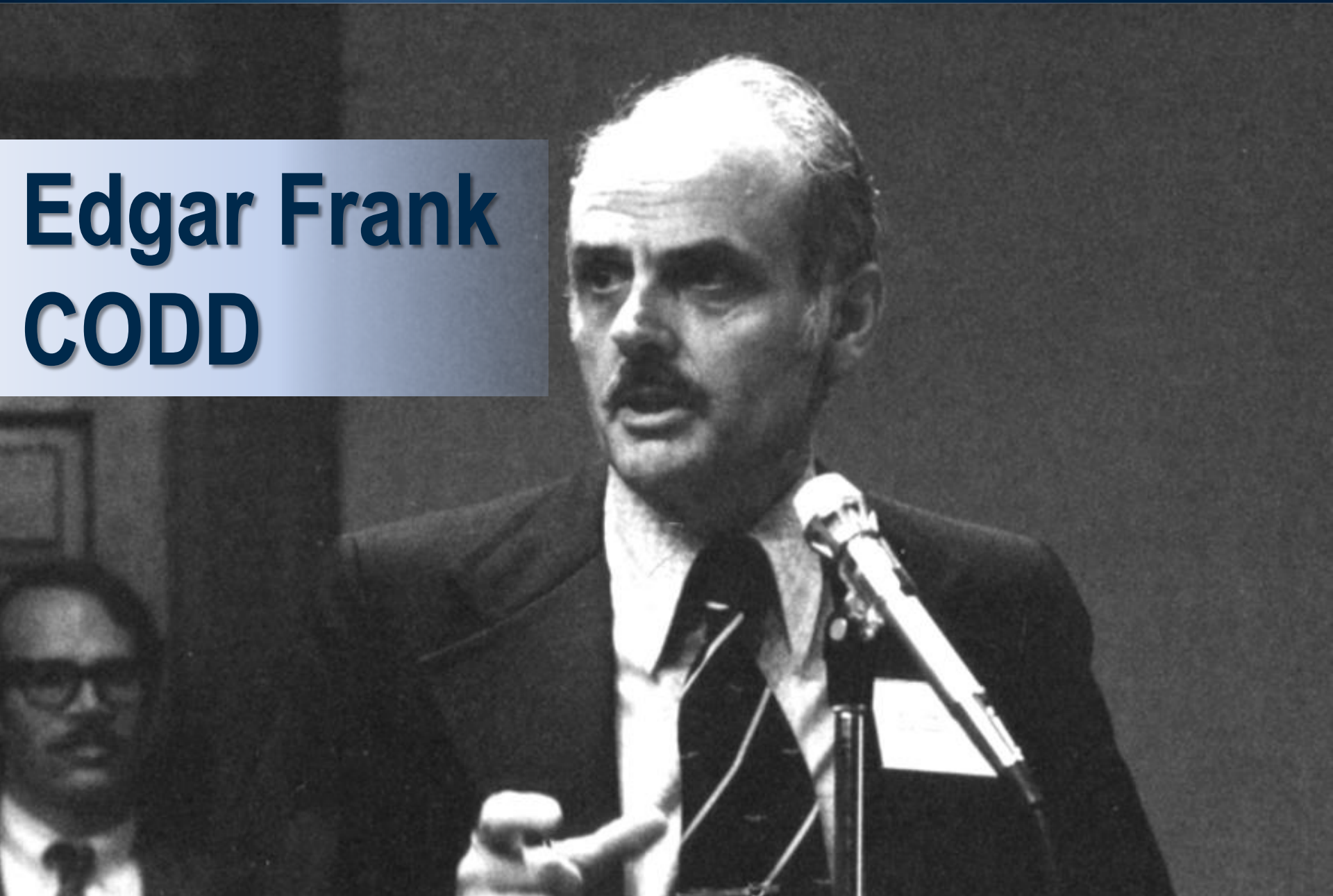
Les SGBD multi-dimensionnels

Les HyperCubes



Les SGBD multi-dimensionnels

**Edgar Frank
Codd**

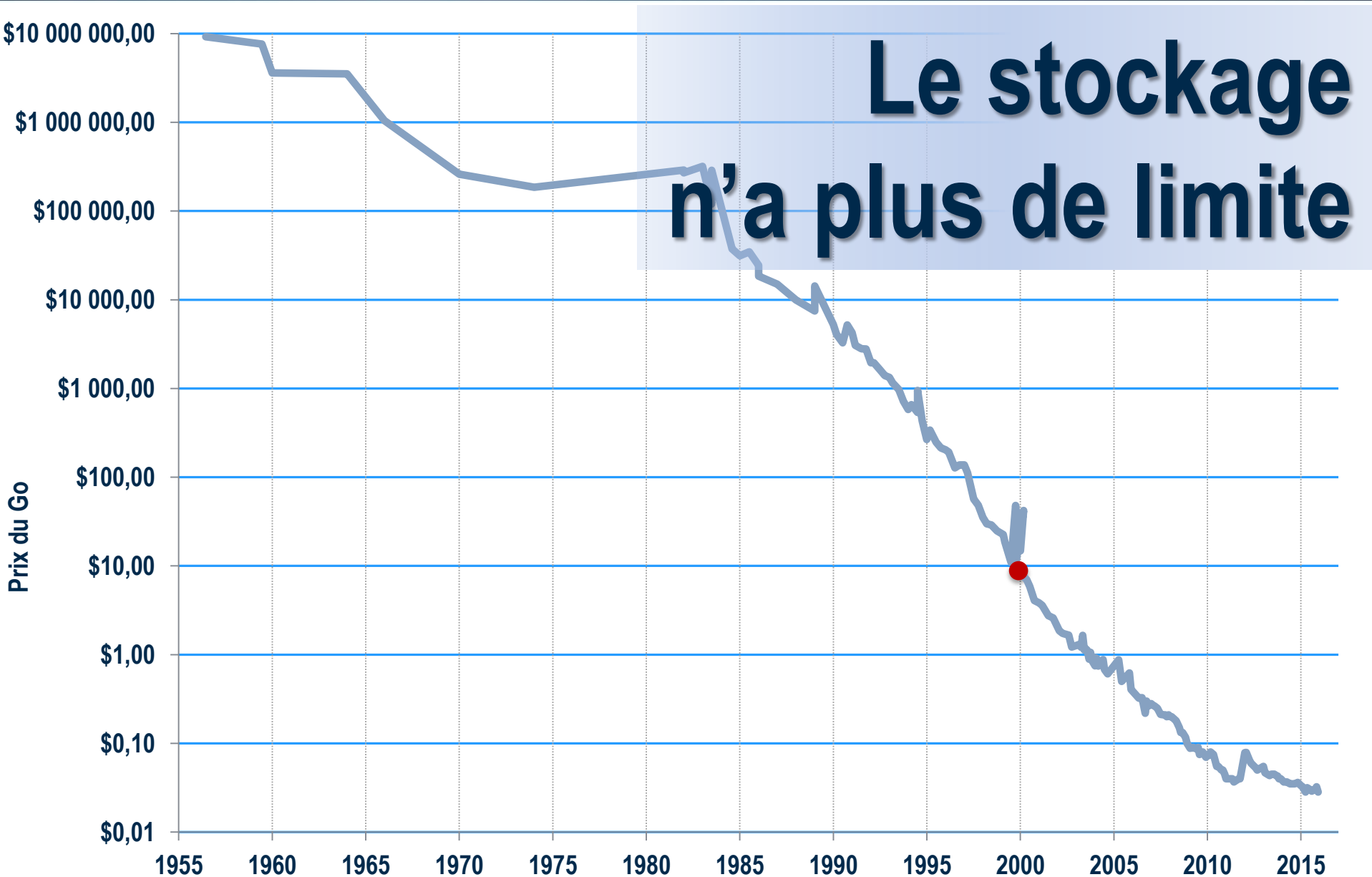


Les SGBD multi-dimensionnels

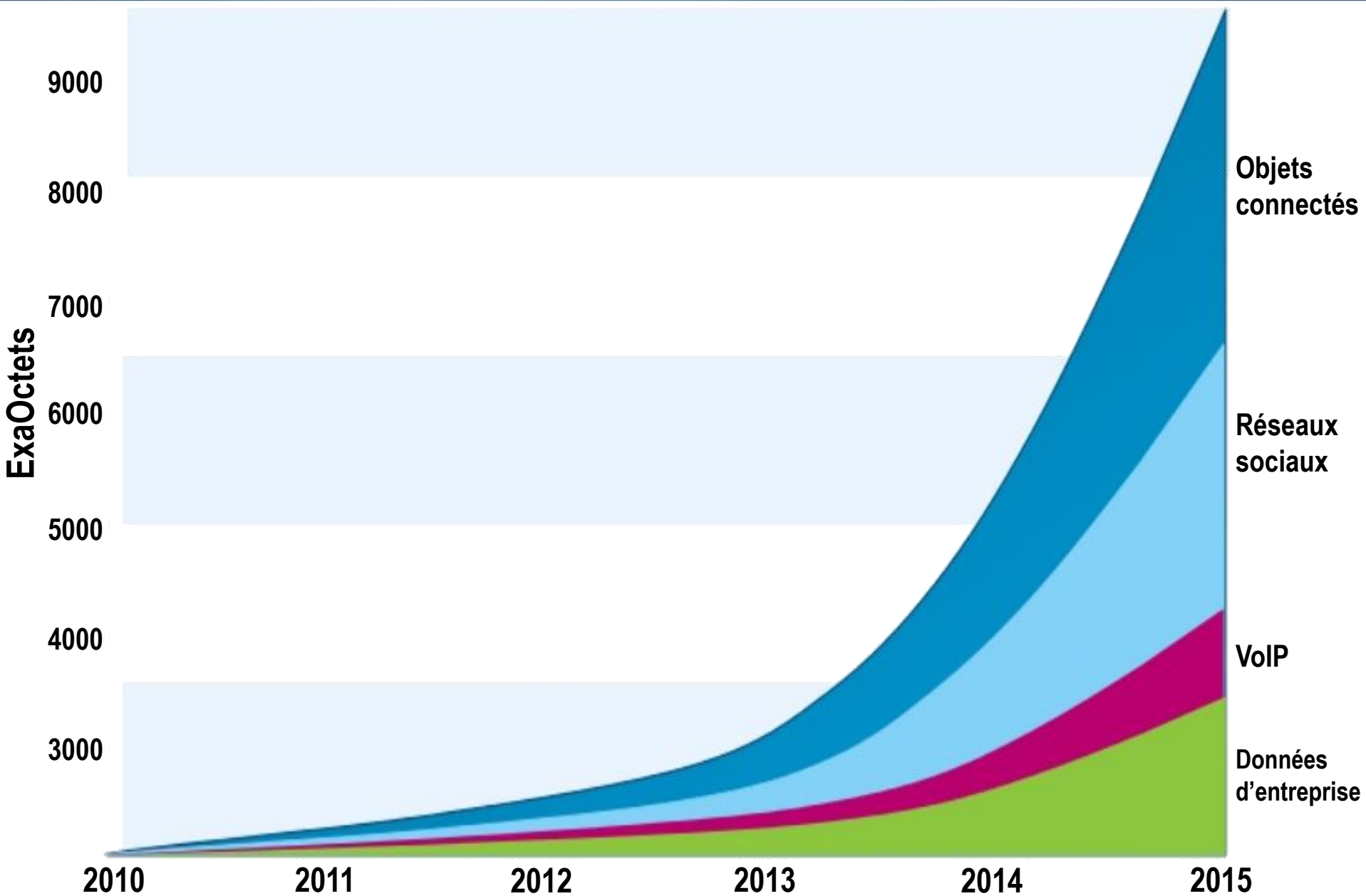
🕒 Les 12 règles de CODD

1. Vue conceptuelle multidimensionnelle
2. Transparence
3. Accessibilité
4. Constance des temps de réponses
5. Architecture client-serveur
6. Indépendance des dimensions
7. Gestion des matrices creuses
8. Accès multi-utilisateurs
9. Pas de restrictions sur les opérations inter et intra dimensions
10. Manipulation aisée des données
11. Simplicité des rapports
12. Nombre illimité de dimensions et nombre illimité d'éléments sur les dimensions

Le XXIème siècle



Le mouvement Big Data



The Google logo is displayed in its characteristic multi-colored font. The 'G' is blue, the first 'o' is red, the second 'o' is yellow, the 'g' is blue, the 'l' is green, and the 'e' is red. The logo is centered horizontally on a white background.



at&t

Le mouvement Big Data

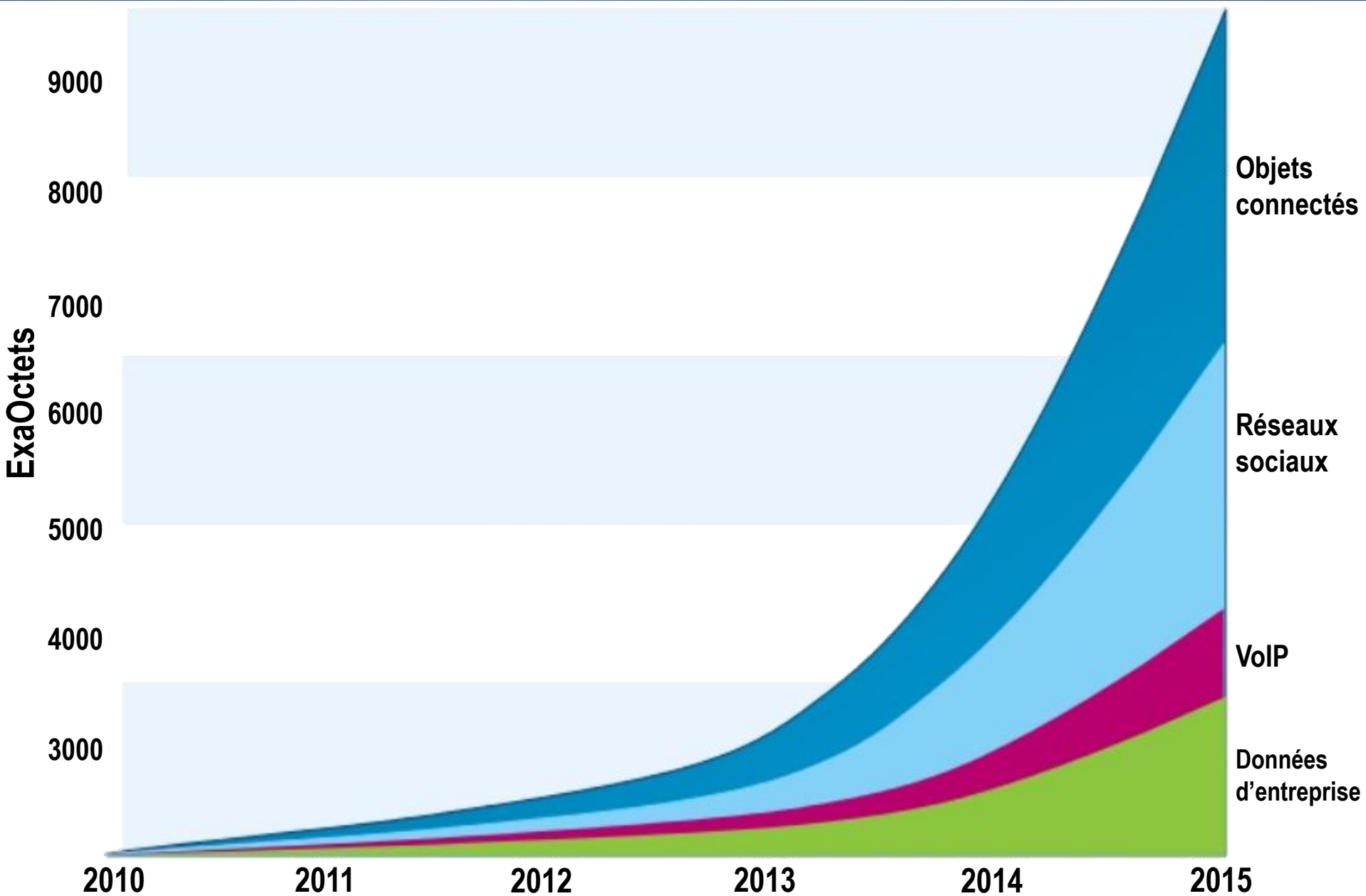
Minority Report



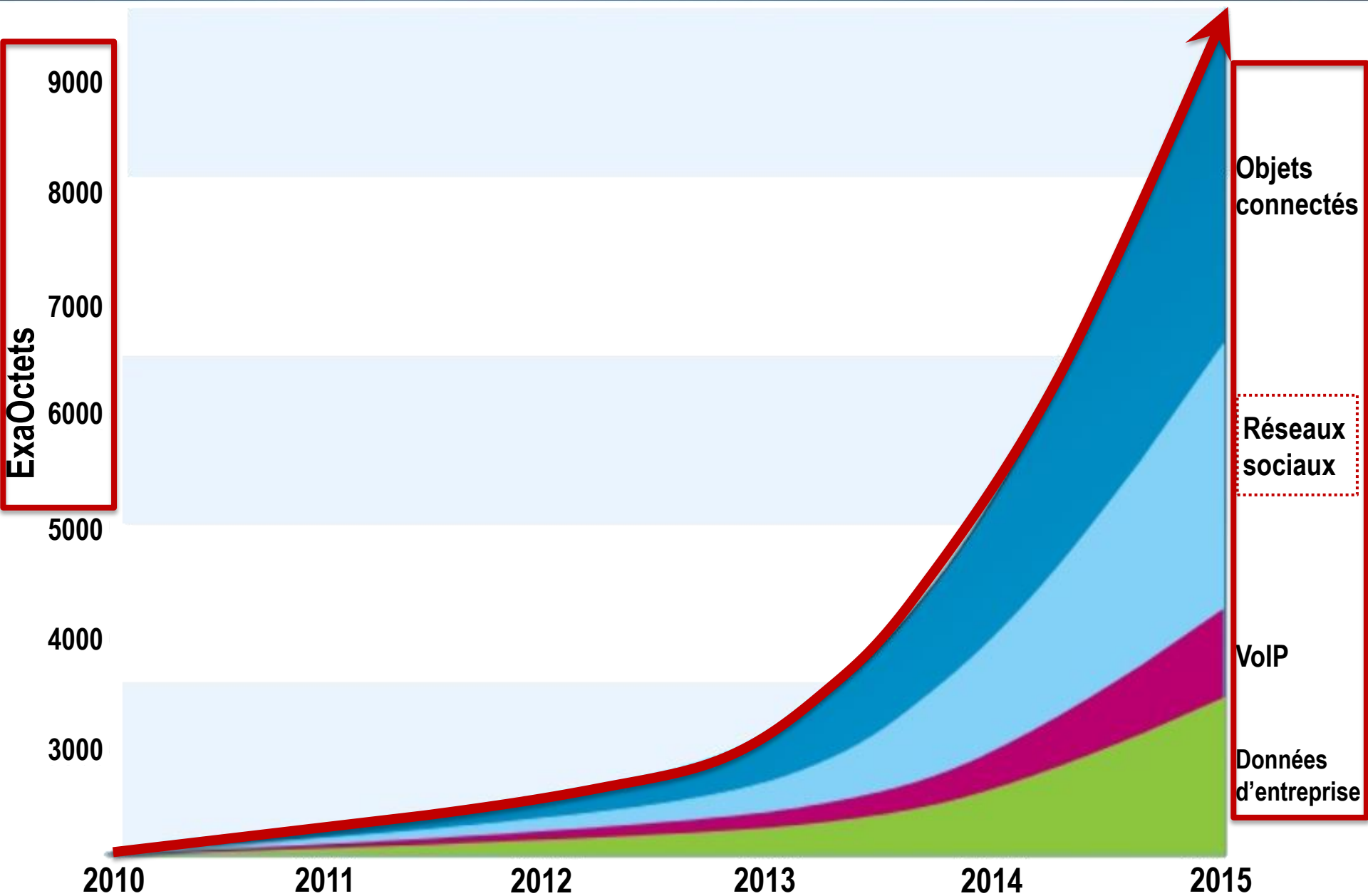
Le mouvement Big Data



Le mouvement Big Data



Le mouvement Big Data



Le mouvement Big Data

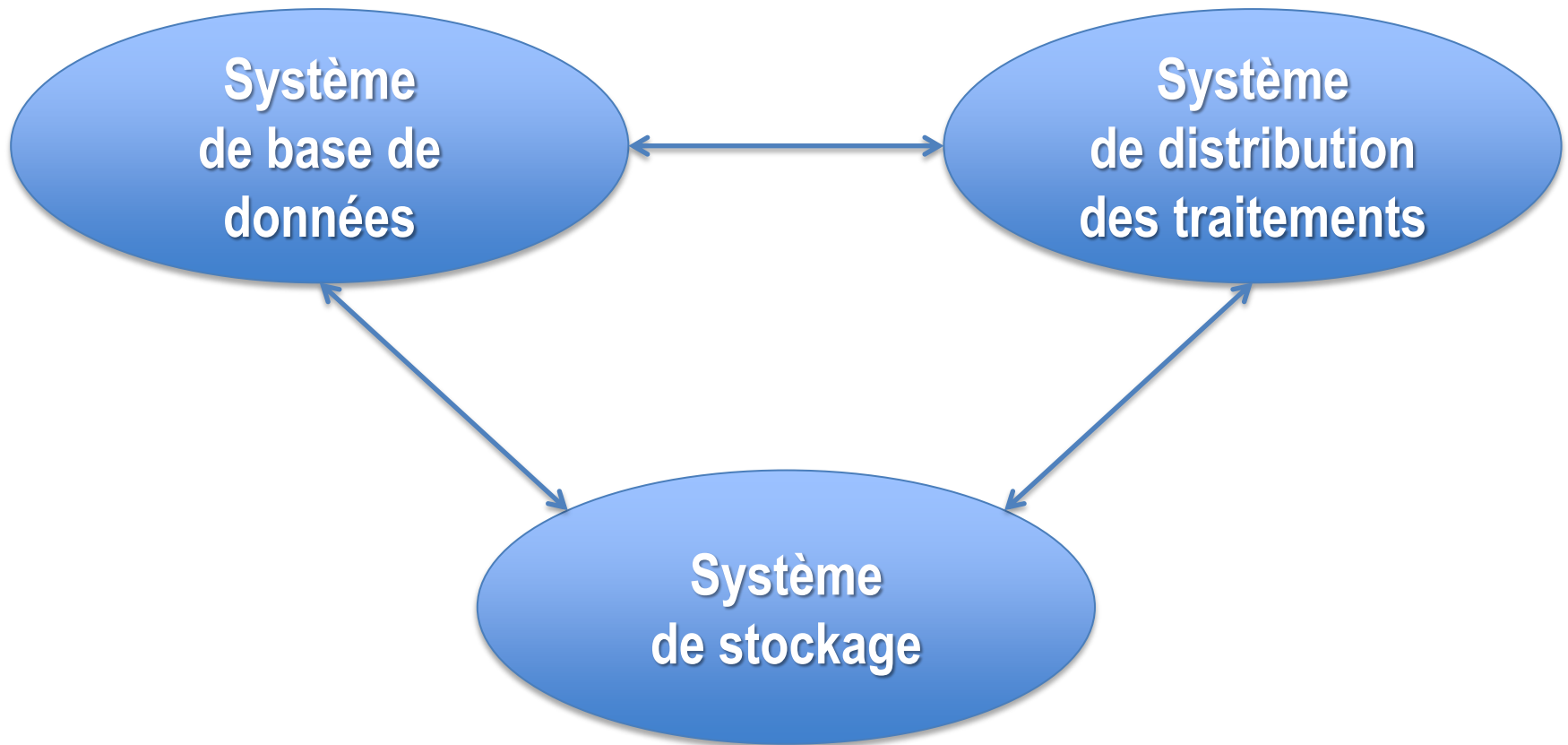
⦿ Loi des 3 V

(<http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf>)

1. **Volume**
2. **Variété**
3. **Vélocité**
4. ***Variabilité***
5. ***Véracité***

Le mouvement Big Data

- ◎ Big data nécessite une architecture distribuée





Le mouvement NoSQL



- ⦿ Les débuts de NoSQL
- ⦿ Qu'est-ce qu'une base de données NoSQL ?
- ⦿ Architecture distribuée
- ⦿ Le théorème CAP
- ⦿ Map Reduce

Les débuts de NoSQL

- ⦿ **Besoin de s'affranchir des propriétés ACID**
- ⦿ **Besoin de flexibilité des modèles de données**
- ⦿ **Besoin d'un système simple à déployer**
- ⦿ **Besoin de s'affranchir des profils DBA ou Admin Système**

Les débuts de NoSQL

The Google logo is displayed in its characteristic multi-colored font. The letters are 'G' (blue), 'o' (red), 'o' (yellow), 'g' (blue), 'l' (green), and 'e' (red). The logo is centered on a white background.

Les débuts de NoSQL

🕒 Google

- Offre variée (Search, Drive, Mail, Maps, ...)
- 2003 : Stockage GoogleFS
- 2004 : Map Reduce
- 2006 : BigTable

🕒 Hadoop

- Distribution de traitement pour le moteur d'indexation Lucene
- Implémentation libre de Map Reduce
- Système de fichier HDFS inspiré de GFS
- Base de données HBase

Qu'est-ce qu'une base de données NoSQL ?

Not Only SQL !

- ⦿ Bases de données NON relationnelles

- ⦿ Flexibilité

 - Données semi ou non structurées

 - ou

 - Structure changeante

- ⦿ Scalabilité

- ⦿ Architecture distribuée

Architecture distribuée

Partitionnement horizontal sur plusieurs nœuds

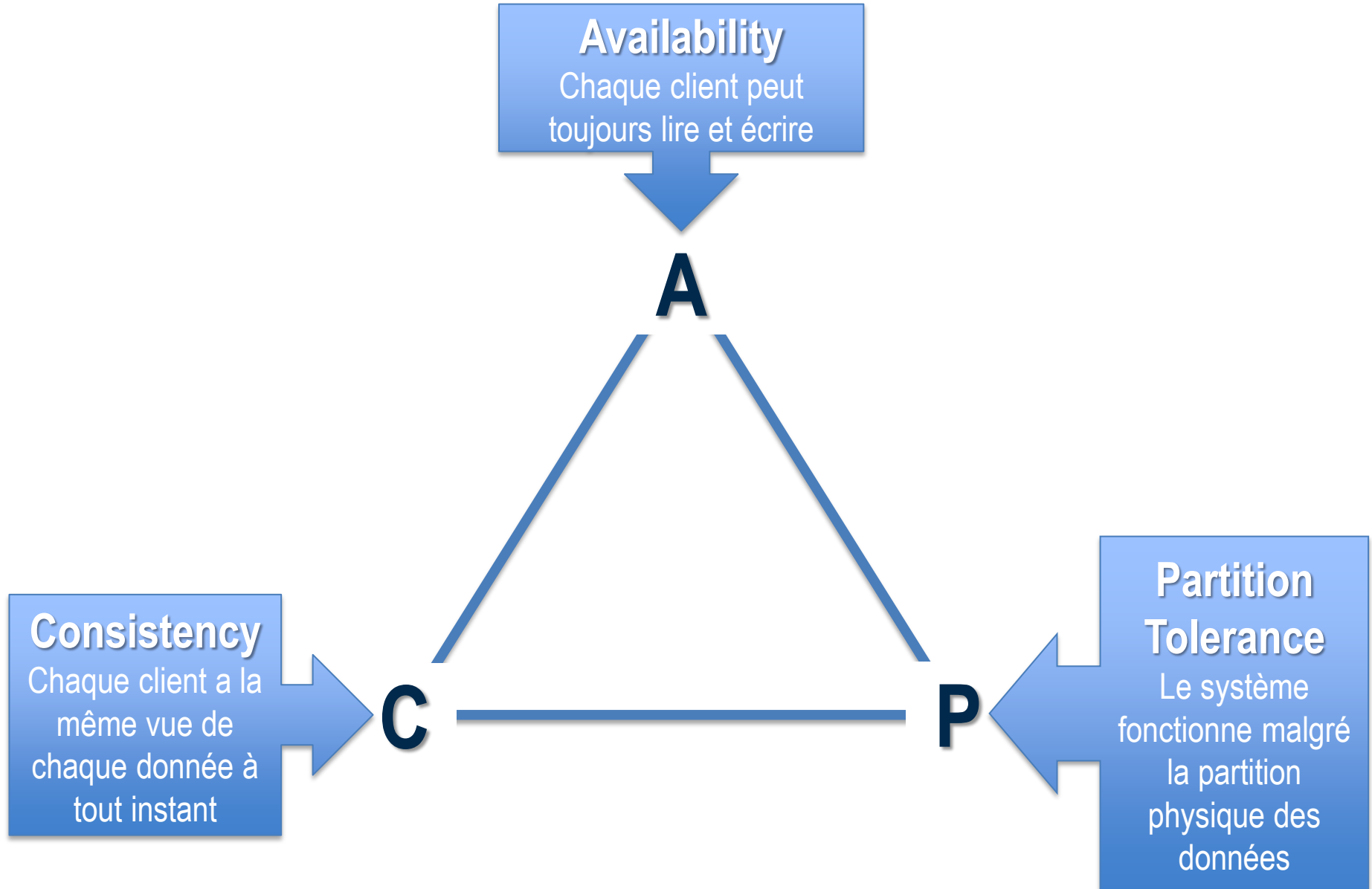
⦿ Distribution avec maître

- Réplication Maître-Esclave
- Sharding
 - L'éclatement est géré automatiquement par le système
 - L'éclatement est obligatoirement distribué
 - La répartition de charge est possible

⦿ Distribution sans maître

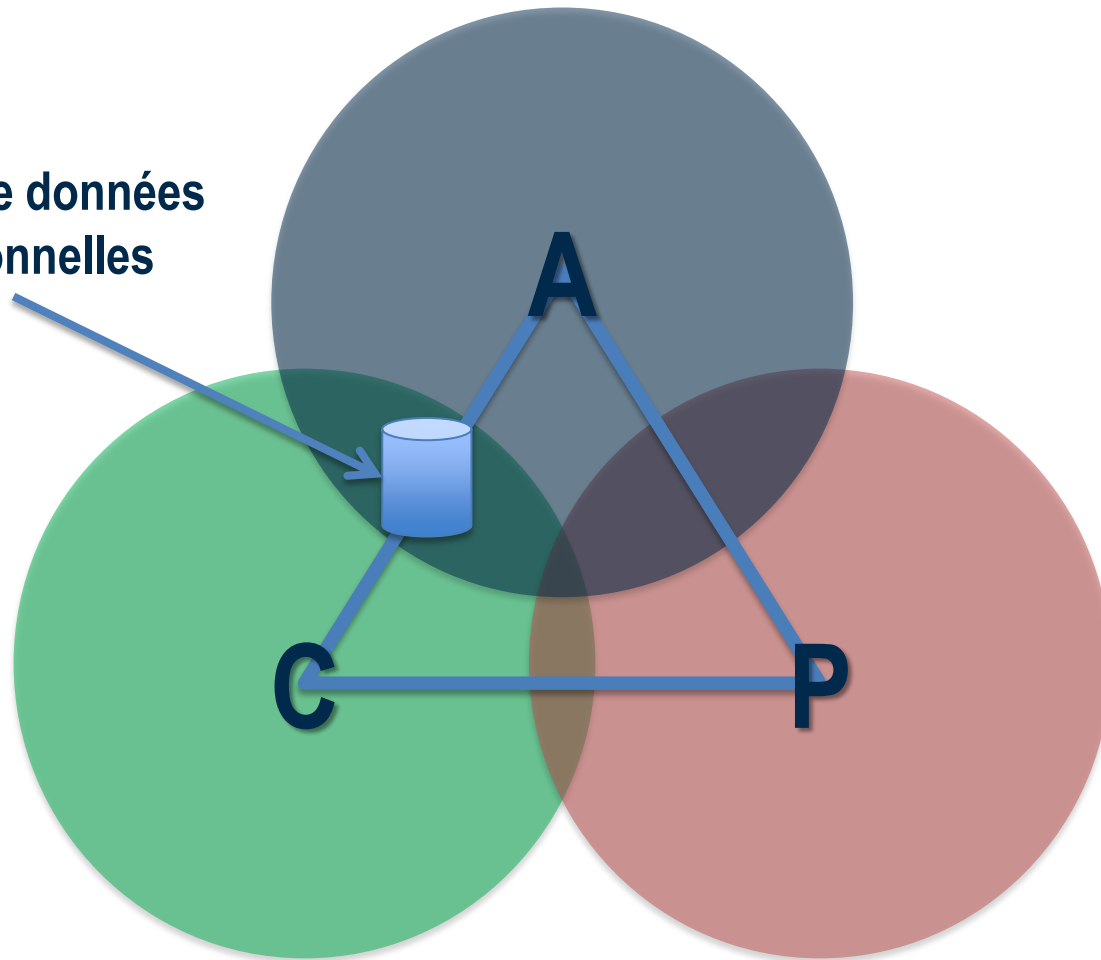
- Réplication par bavardage
- Répartition par distribution des clés

Le théorème CAP



Le théorème CAP

Bases de données
relationnelles





Les grandes familles de bases de données NoSQL



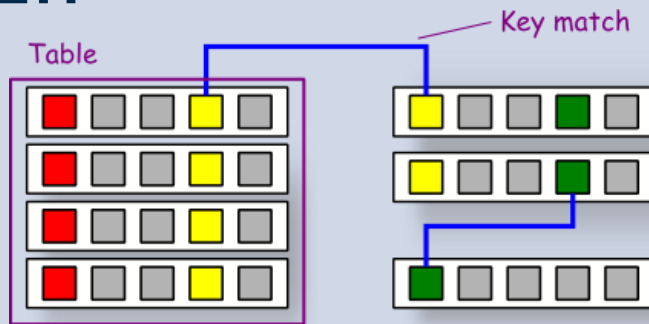
- ⦿ Les grandes familles de bases de données
- ⦿ Les 4 grandes familles NoSQL

Les grandes familles de bases de données

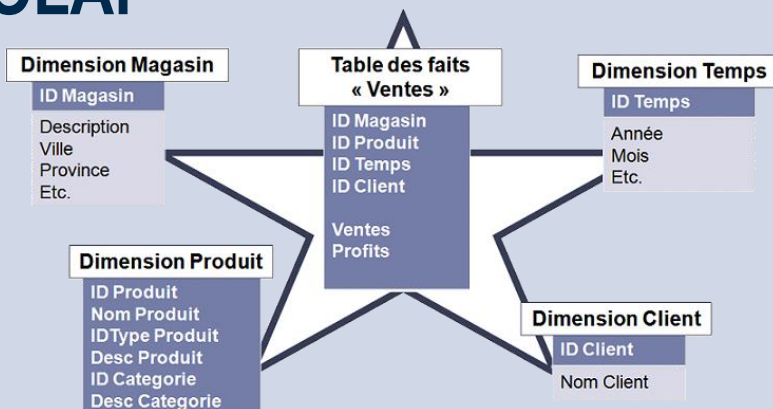
SQL

Non SQL

OLTP



OLAP



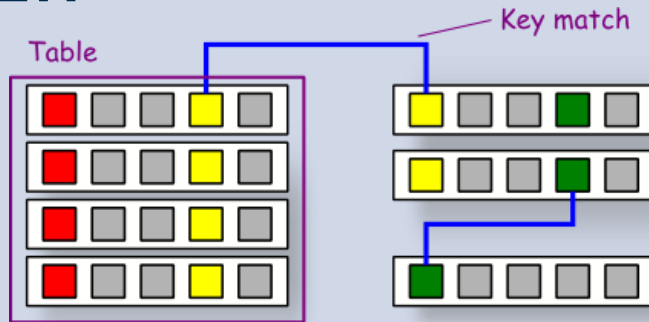
225 SGBD

<http://nosql-database.org/>

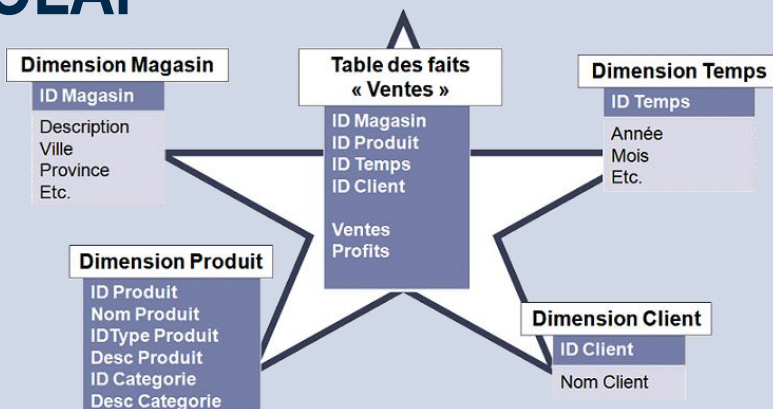
Les grandes familles de bases de données

SQL

OLTP



OLAP



Non SQL

Clé-valeur

Document

Colonnes

Graphe

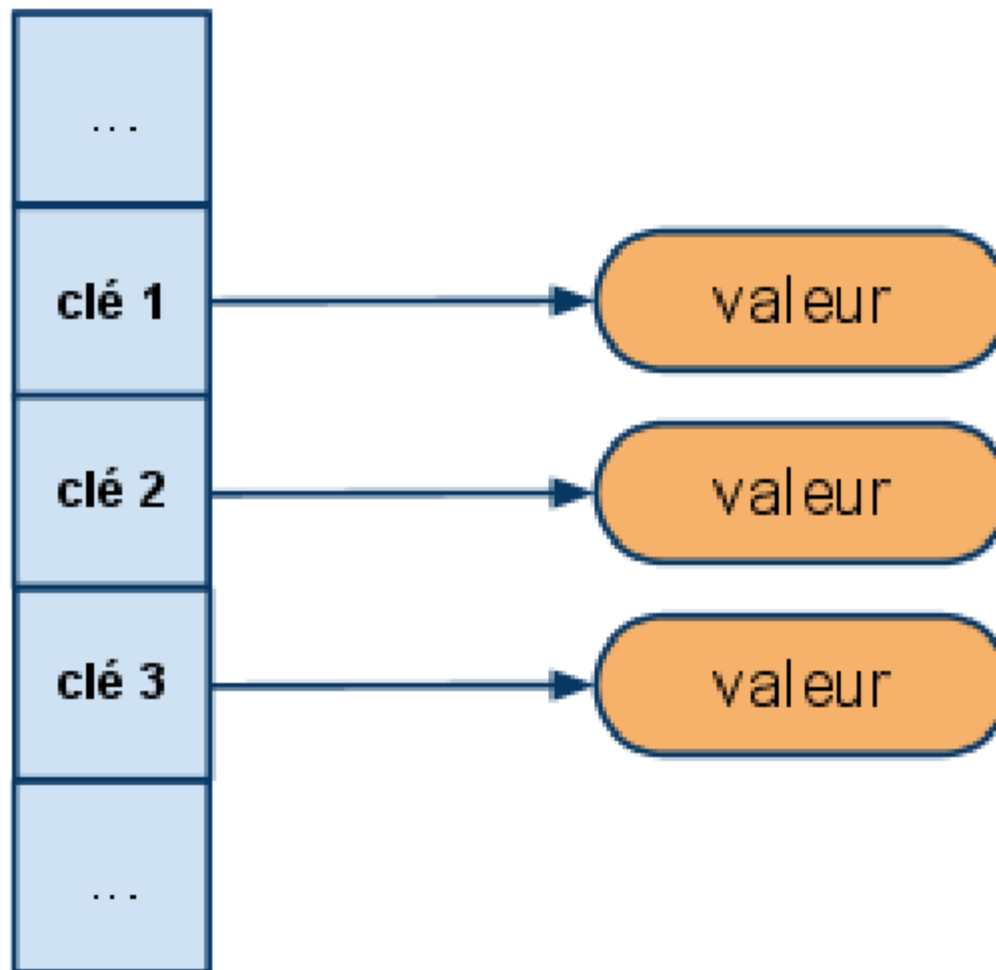


Les 4 grandes familles du NoSQL



- ⦿ Clés-valeurs
- ⦿ Document
- ⦿ Orientées colonnes
- ⦿ Graphes

Clés-valeurs



Clés-valeurs



Clés-valeurs

🕒 1 base = 1 table

Clés primaires	Valeurs
Loto : Samedi 23 avril 2016	2, 15, 17, 22, 26 - 7
La grande question sur la vie, l'univers et le reste	42
logo cnrs	
http://prodev.cnrs.fr/doku.php?id=nosql2016	<pre><html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr" dir="ltr" class="no-js"> <head> <meta charset="UTF-8" /> <title>nosql2016 [prodev]</title> ... <body> ... </body> </html></pre>

Clés-valeurs

⦿ API simple

- Put
- Get
- Delete
- *Update (facultatif)*

Clés-valeurs

🕒 Exemple Twitter

○ Utilisateurs

Clés primaires	Valeurs
user:guillaumeharry:nom	HARRY
user:guillaumeharry:prenom	guillaume

○ Messages

Clés primaires	Valeurs
message:post32	"user_id":"guillaumeharry", "time":"26/04/2016", "body":"En pleine présentation"
message:post33	"user_id":"bernardchetrit", "time":"26/04/2016", "body":"En pleine concentration"

○ Mots clés

Clés primaires	Valeurs
pleine	post32, post33
concentration	post33

Clés-valeurs

🕒 Exemple Twitter

○ Contenu de la base de données

Clés primaires	Valeurs
user:guillaumeharry:nom	HARRY
user:guillaumeharry:prenom	guillaume
message:post32	"user_id":"guillaumeharry", "time":"26/04/2016", "body":"En pleine présentation"
message:post33	"user_id":"bernardchetrit", "time":"26/04/2016", "body":"En pleine concentration"
pleine	post32, post33
concentration	post33
tion	présentation, concentration
pre	présentation

Clés-valeurs

🎯 Cible

- Système de cache (memcache)
- Stockage de gros volume de données
- Collecte d'évènements

🎯 Avantage

- Recherche rapide
- Table à 2 colonnes

🎯 Inconvénient

- Aucun schéma
- Pas de requête possible sur les valeurs nativement
- Pas de garantie d'intégrité

Clés-valeur

Implémentations

⦿ **Dynamo**

- Développée en 2007 et utilisée par Amazon pour gérer le panier d'achat

⦿ **Voldemort**

- Développée et utilisée par LinkedIn

⦿ **Riak**

- Edité par Basho Technologies
- Inspirée de **Dynamo**
- Base hybride orientée clé/valeur ou document

⦿ **Redis**

- Hautement performant
- Redis n'est pas tolérant aux pannes

⦿ **Memcached**

- Cache mémoire distribué
- Epruvé depuis de nombreuses années

Clés-valeur

En production

- ① **The Guardian**

(quotidien d'information britannique)

- ② **GitHub**

- ③ **Stack Overflow**

- ④ **Craigslist**

(site web américain de petites annonces et de forums de discussion)

- ⑤ **YouPorn**

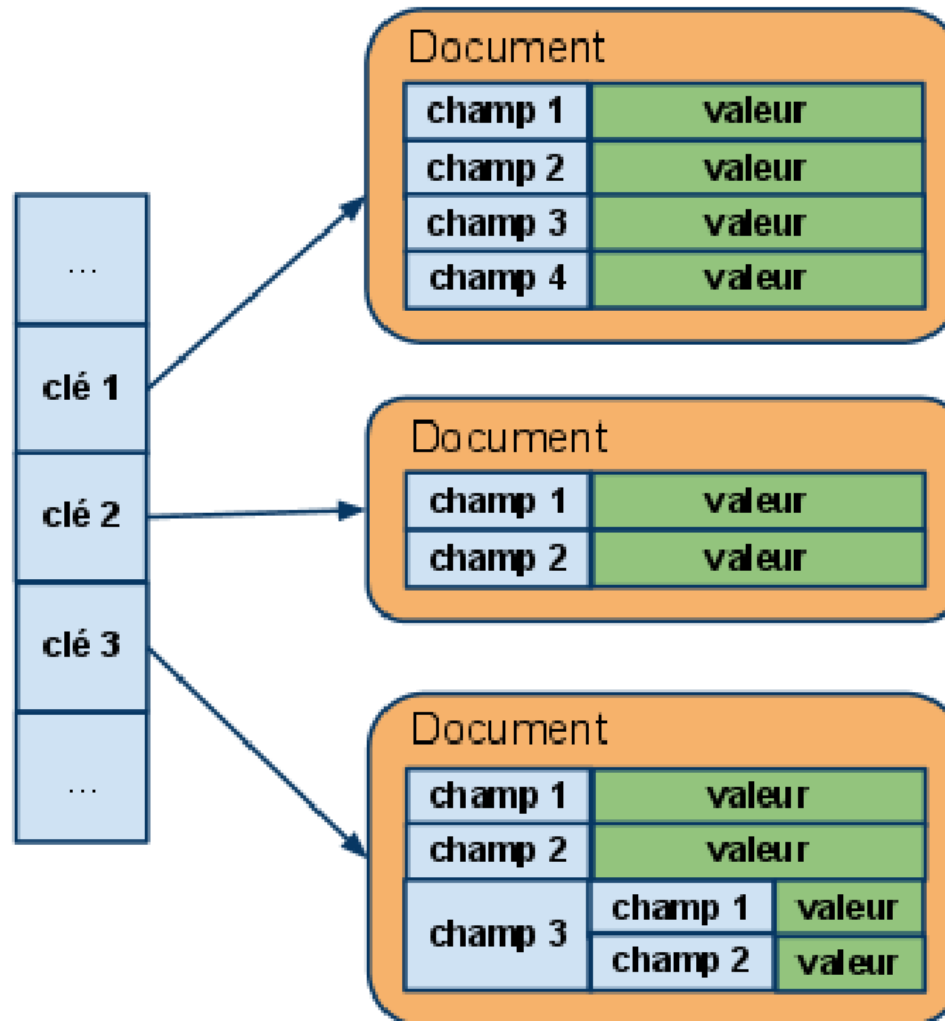


Les 4 grandes familles du NoSQL



- ⦿ Clés-valeurs
- ⦿ **Document**
- ⦿ Orientées colonnes
- ⦿ Graphes

Document



Document

⦿ Proche du système clé-valeur

⦿ Base de données

- Ensemble de collections

⦿ Collection

- Ensemble de documents
- Equivalent aux tables dans les SGBDR

⦿ Document

- Unité de base de stockage
- Equivalent aux lignes dans les SGBDR
- Ensemble de couples clé-valeur au format JSON ou XML
- Chaque document possède un identifiant unique dans une collection

Document

Exemple

```
{
  "_id" : ObjectId("56a01213f89947e35a4a6d01"),
  "name" : "mycom",
  "description" : {
    "title" : "Conditions générales d'utilisation de My Com",
    "summary" : "Demande d'activation du service My Com après acceptation des conditions générales d'utilisation",
    "content" : "\r\nCGU My COM\r\nOffre de Service\r\nV1.0 Janvier 2016\r\n\r\nI.\tObjet\r\nLes présentes conditions générales d'utilisation (Dénommées ci-après CGU",
    "url" : "../docs/My_Com_CGU_v1.0.pdf"
  },
  "version" : "1.0",
  "fromDate" : "2015-12-31T10:00",
  "toDate" : "2015-12-31T10:00",
  "type" : "GTU",
  "dialogMessage" : "Merci d'avoir signé les CGU de My Com. Votre demande de création de compte a bien été enregistrée. Vous allez recevoir un e-mail quand cette créat",
  "linkedRequest" : {
    "name" : "mycom",
    "type" : "MAILING_LIST",
    "version" : "1.0"
  },
  "decisionMessage" : "J'accepte les conditions générales d'utilisation du service"
}

{
  "_id" : ObjectId("56a14957f89987311256e59f"),
  "name" : "mycom",
  "description" : {
    "title" : "Liste de diffusion My Com",
    "summary" : "Inscription à la liste de diffusion de My Com",
    "content" : "Votre inscription sur notre liste de diffusion nous permettra de vous \ntenir informé des actualités du service : \n - indisponibilités pour cause d",
  },
  "version" : "1.0",
  "type" : "MAILING_LIST",
  "dialogMessage" : "Merci d'avoir signé les CGU de My Com. Votre demande de création de compte a bien été enregistrée. Vous allez recevoir un e-mail quand cette créat",
  "linkedToParent" : true,
  "decisionMessage" : "J'accepte d'être inscrit à la liste de diffusion My Com"
}
```

Document

```
{
  "_id": "548855bb3c5f16d690df3be6",
  "infos": {
    "firstName": "John",
    "lastName": "DOE",
    "uid": "john.doe.2",
    "mail": "john.doe@cnrs.fr",
    "idEbooking": "8GEB8AAF",
    "unit": {
      "code": "UMR1234",
      "label": "Unite mixte de travail sur des choses tres interessantes",
      "acronym": "UMTCI",
      "implantationNumber": "123456789"
    },
    "mainDr": {
      "code": "05",
      "label": "Ile de France - Ouest et Nord",
      "supportUnit": "MOY0500"
    },
    "billingCenter": {
      "code": "05",
      "label": "Ile de France - Ouest et Nord",
      "supportUnit": "MOY0500",
      "modifiedByAdmin": false
    },
    "functions": [
      {
        "domain": "CNRS",
        "code": "DU",
        "value": "UMR5678"
      },
      {
        "domain": "INRIA",
        "code": "ABC",
        "value": "ppp_34jjh"
      }
    ],
    "present": true,
    "locked": false,
    "birthDate": "1974-12-10"
  },
  "cards": {
    "4d74fbe0-ab16-11e4-8f72-0002a5d5c51b": {
      "orderID": 1,
      "type": "PASSPORT",
      "number": "34-ERER-DFDFDF",
      "country": "NO",
      "city": "Tromsø",
      "submissionDate": "1998-01-04",
      "expirationDate": "2008-01-04"
    }
  },
  "addresses": {
    "professional": {
      "phone": "33-(1) 23456789",
      "fax": "33-123456780",
      "mail": "john.doe@cnrs.fr",
      "address": "33 bis rue de la mairie",
      "complement": "4eme etage",
      "postalCode": "2A3245",
      "city": "Perpignan",
      "country": "FR"
    },
    "personal": {
      "phone": "33-567766789",
      "mobile": "33-665544333",
      "address": "1221 avenue d'Italie",
      "postalCode": "2A3246",
      "city": "Carcassonne",
      "country": "FR",
      "otherMail": "johnatan.doe@gmail.com"
    }
  }
},
```

Document

🎯 Cible

- Journalisation d'évènements
- Application CRUD
- Recherche complexe

🎯 Avantage

- Données semi-structurées
- Gestion de la version du document

🎯 Inconvénient

- Performances des requêtes

Document

Implémentations

⦿ CouchDB

- Pas de verrou lors des accès concurrents

⦿ MongoDB

- Développé par la société 10gen
- Permet d'indexer les propriétés des documents afin d'optimiser une recherche
- Support de l'indexation géospatiale

⦿ Riak

- Base hybride orientée clé/valeur ou document

Document

En production

🕒 Ebay

Métadonnées de chaque objet vendu

🕒 Expedia

🕒 McAfee

Référentiel central pour la plateforme de détection des menaces

🕒 City of Chicago

Données géospatiales pour gérer toutes les urgences (situation des bus, 911, ...)

🕒 Telefonica

🕒 Marriott

🕒 PayPal

🕒 Ryanair

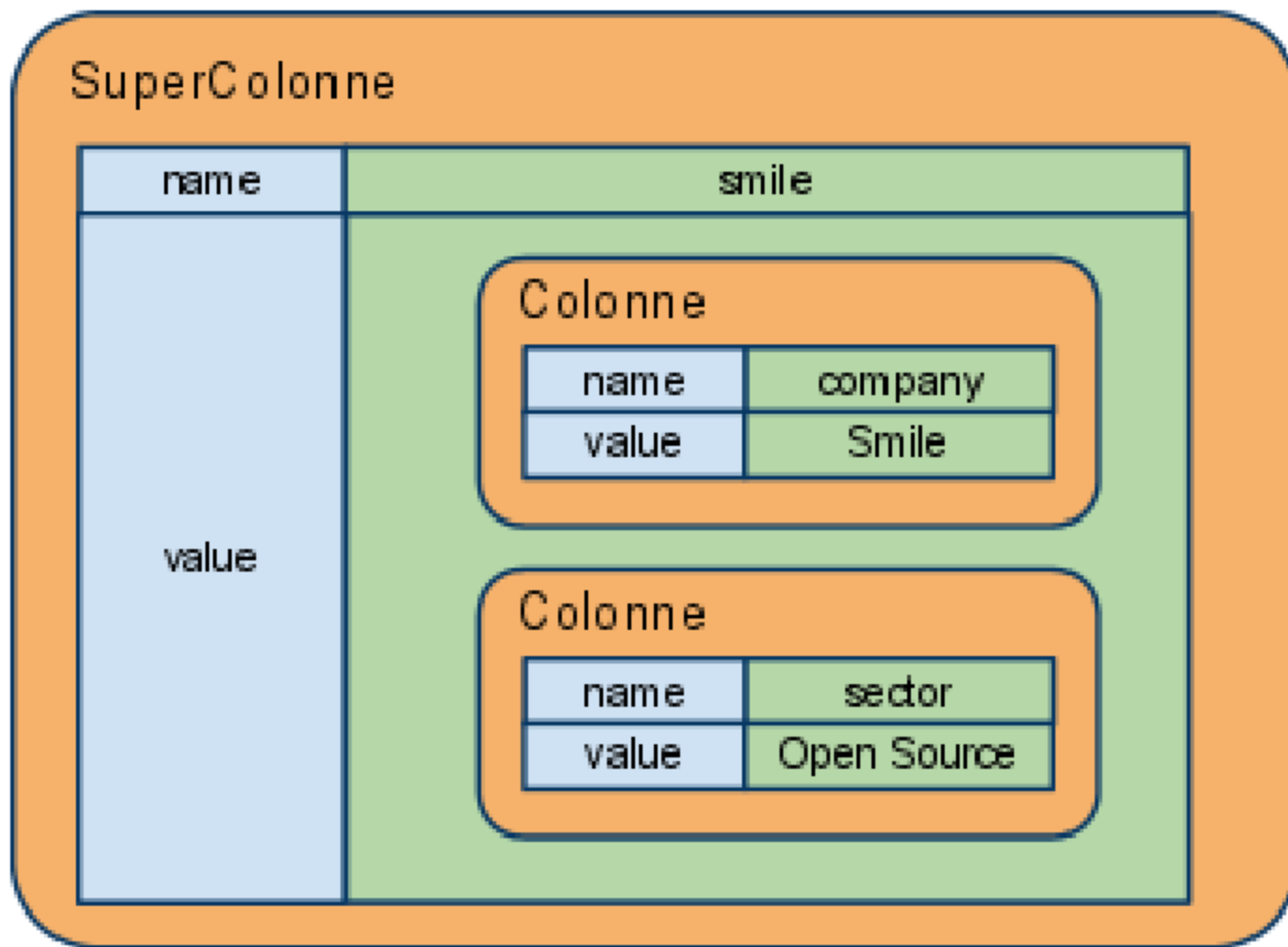


Les 4 grandes familles du NoSQL



- ⦿ Clés-valeurs
- ⦿ Document
- ⦿ **Orientées colonnes**
- ⦿ Graphes

Orientées colonnes



Orientées colonnes

● Famille de colonnes

- Equivalent à une table dans une base de données relationnelle

Clé de ligne	Nom1 Valeur1	Nom2 Valeur2	Nom3 Valeur3	Nomx Valeurx
Clé de ligne	Nom1 Valeur1		Nom3 Valeur3	Nomx Valeurx
Clé de ligne	Nom1 Valeur1	Nom2 Valeur2	Nom3 Valeur3	Nomx Valeurx

Orientées colonnes

⦿ Structure des données par familles de colonnes

- 1 clé pour accéder à un ensemble de colonnes
- Colonnes sont groupées par famille de colonne

⦿ Données stockées par colonne

- Chaque colonne est définie par un couple clé-valeur
- Les colonnes sont regroupées par ligne
- Chaque ligne est identifiée par un identifiant unique.

⦿ API de (très) bas niveau

⦿ Requête sur

- Lignes
- Familles de colonnes
- Noms de colonnes

Orientées colonnes

● La colonne est une valeur !

○ Nom de colonne

Tous les trains qui passent à Valence

train6101	PARIS 06:07	VALENCE 08:19	AVIGNON 09:07	AIX 09:30	MARSEILLE 09:41
train2917	AVIGNON 11:30				11:59 MARSEILLE

○ Tri sur les colonnes (slice)

Toutes les données de 9h à 11h

capteur1	08:55 123	09:00	...	11:00 52	11:05 19
----------	--------------	-------	-----	-------------	-------------



● MAIS PAS DE JOINTURE !

Orientées colonnes

🎯 Cible

- Répond aux problématiques
 - de charge
 - de volume
 - de très haute disponibilité
- Timeseries
 - Données stockées suivant des timestamps
 - Adapter aux données issues de capteurs
- Clickstreams
 - Enregistrement des actions d'un utilisateur sur une application

🎯 Avantages

- Forte tolérance aux pannes
- Facilite l'agrégation

🎯 Inconvénient

- API de (très) bas niveau

Orientées colonnes

Implémentations

⦿ Google BigTable

⦿ Hbase

- Clone de **BigTable** développé au sein de l'écosystème Hadoop
- Très performant en lecture

⦿ Cassandra

- Initialement créé par Facebook
- Très performant en écriture
- **Pas de garantie de cohérence stricte**

Orientées colonnes

En production

- Google

- Netflix

Historique complet des visualisations des 36 millions d'utilisateurs

- Ebay

Données sociales (like, own, want)

- Twitter

- Cisco

- Facebook

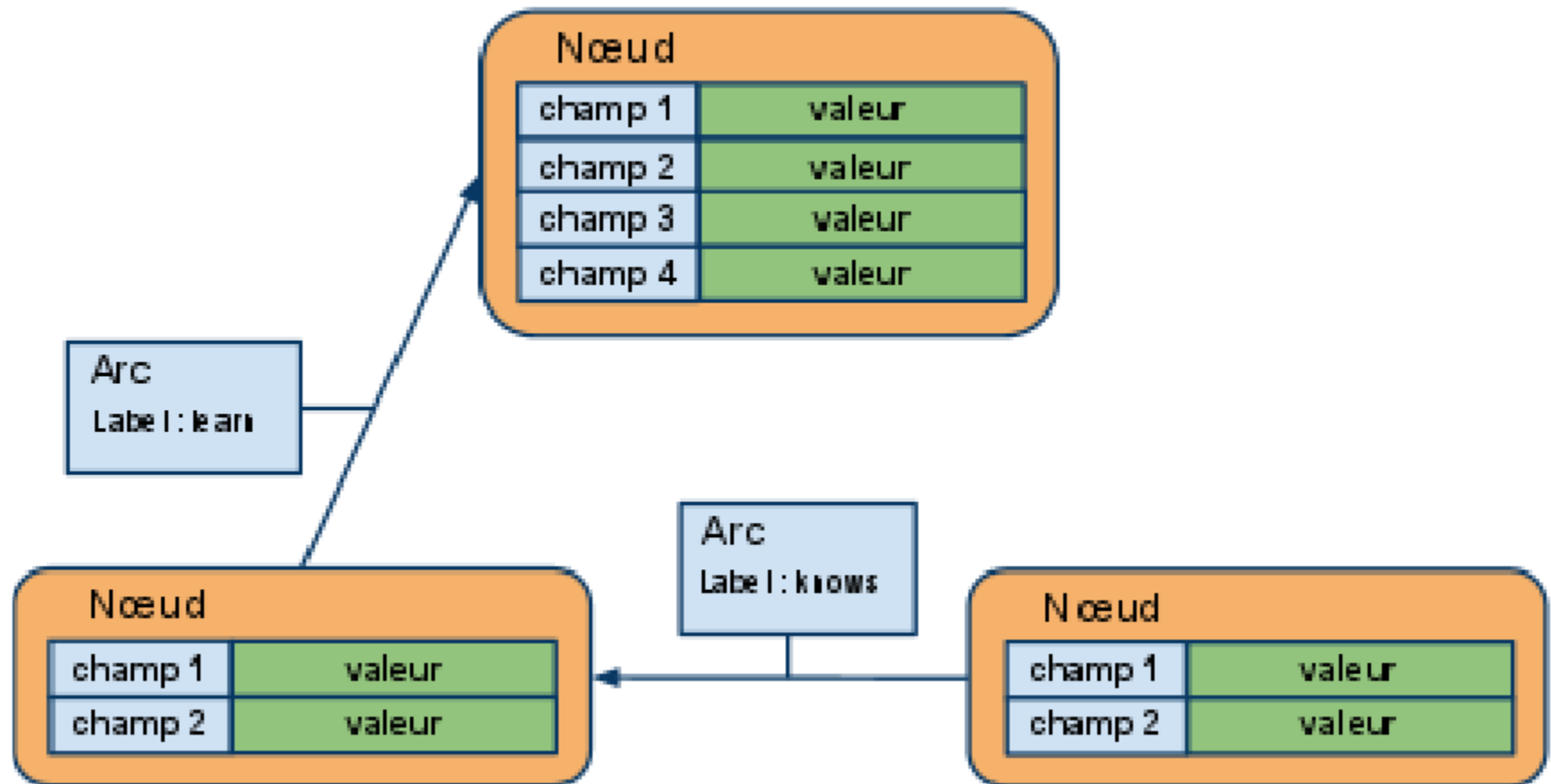


Les 4 grandes familles du NoSQL



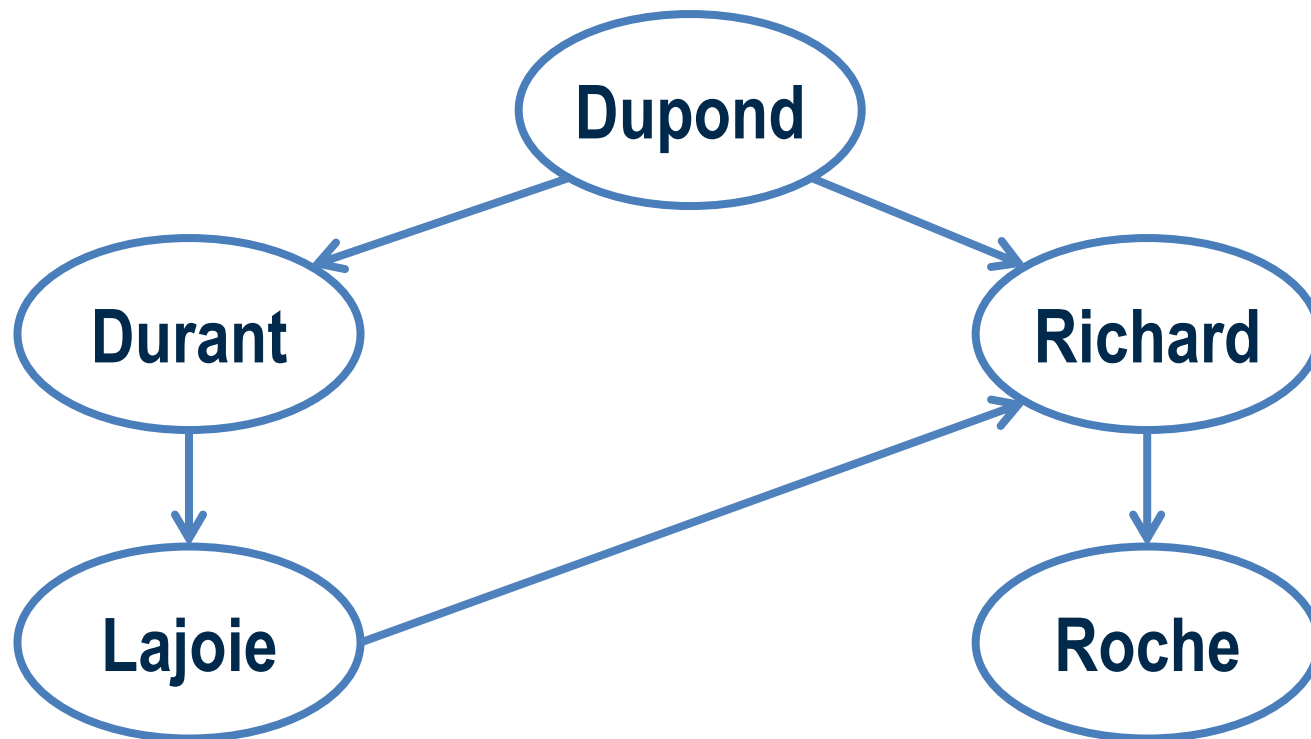
- ⦿ Clés-valeurs
- ⦿ Document
- ⦿ Orientées colonnes
- ⦿ Graphes

Graphe



Graphe

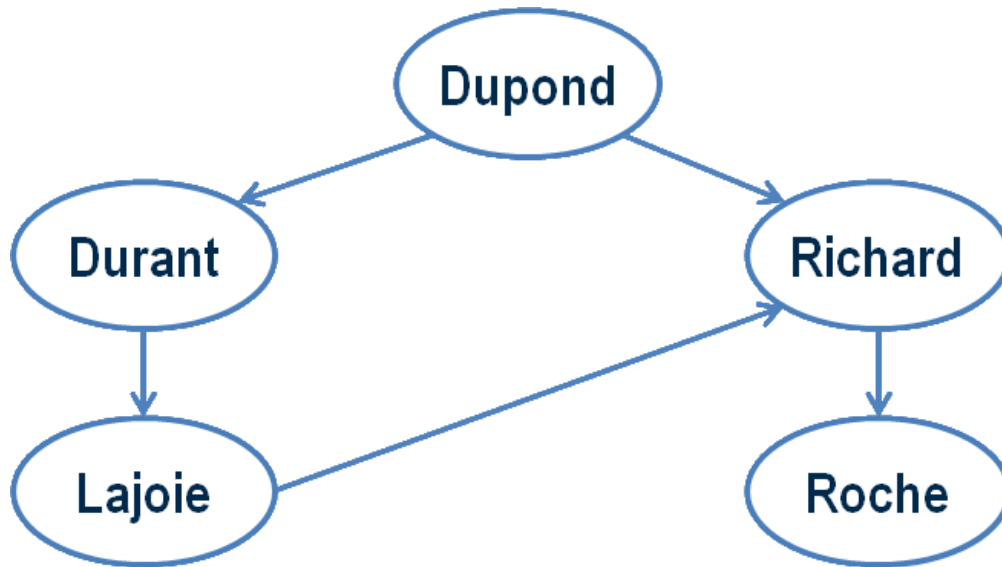
- ⦿ **Permet de décrire les relations entre des entités**
 - Modélisation des problématiques spécifiques avec forte connectivité
 - Recherche de liens optimaux
- ⦿ **Exemple : matérialiser un réseau d'amis**



Graphe

🕒 Exemple : matérialiser un réseau d'amis en SQL

○ 2 tables USER, USER_FRIENDS



ID	NOM
1	Dupond
2	Durand
3	Richard
4	Ducode
5	Roche

ID	FROM_USER	TO_USER
1	1	2
2	1	3
3	2	5
4	5	3
5	3	4

Graphe

🕒 Exemple : matérialiser un réseau d'amis en SQL

○ Compter les amis de 1^{er} niveau

```
SELECT u1.nom, count(*) FROM user u1 INNER JOIN user_friends uf1 WHERE u1.id = uf1.user1 GROUP BY nom
```

nom	count(*)
Dupont	2
Durant	1
Lajoie	1
Richard	1

○ Compter les amis de 2nd niveau

```
select u1.nom, count(*) FROM user AS u1  
  INNER JOIN user_friends AS uf1 on u1.id = uf1.user1  
  INNER JOIN user_friends AS uf2 on uf1.user2=uf2.user1  
  GROUP BY u1.nom
```

nom	count(*)
Dupont	2
Durant	1
Lajoie	1

○ Compter les amis de 3^{ème} niveau

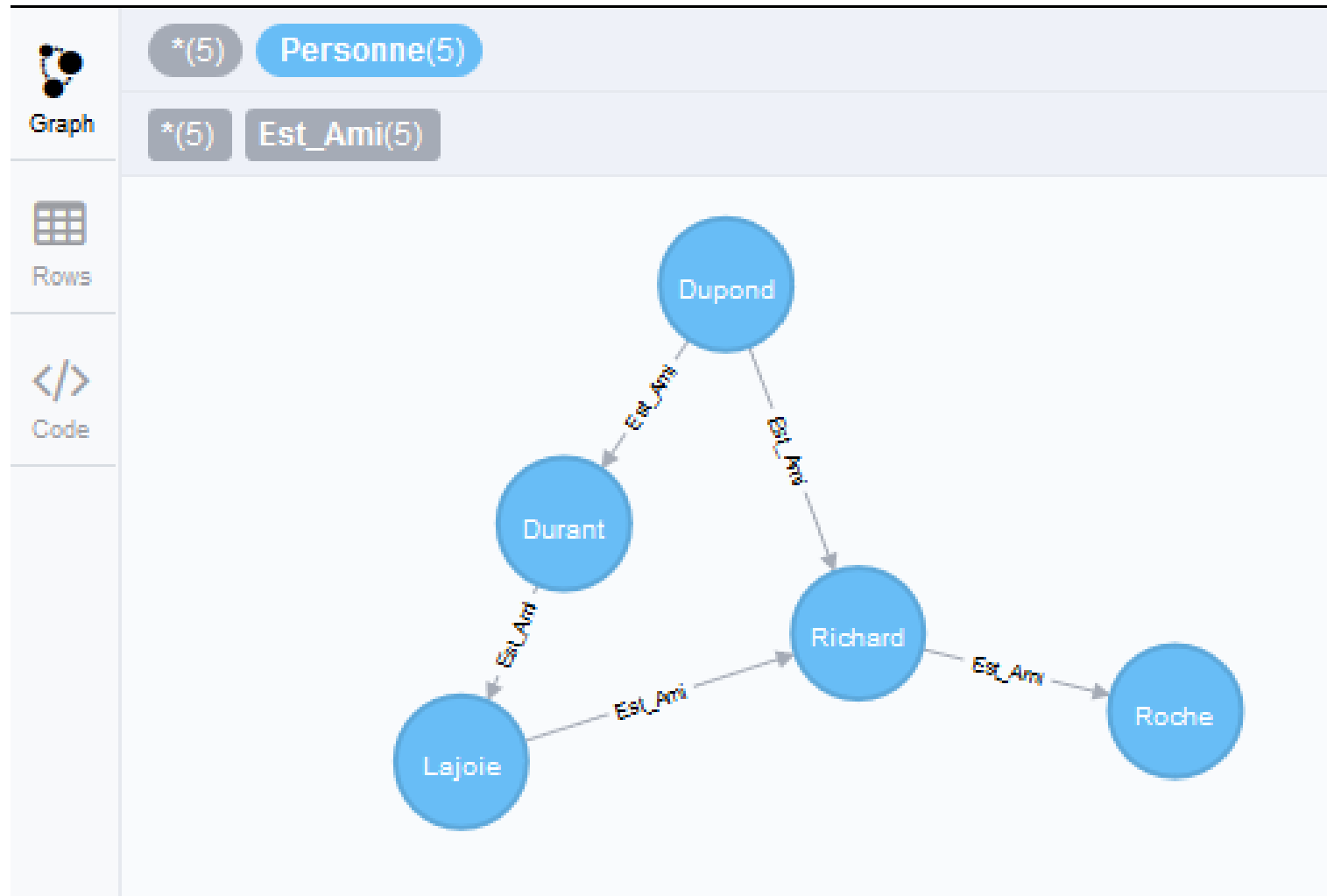
```
select u1.nom, count(*) FROM user AS u1  
  INNER JOIN user_friends AS uf1 on u1.id = uf1.user1  
  INNER JOIN user_friends AS uf2 on uf1.user2=uf2.user1  
  INNER JOIN user_friends AS uf3 on uf2.user2=uf3.user1  
  GROUP BY u1.nom
```

nom	count(*)
Dupont	1
Durant	1

Graphe

🕒 Exemple : matérialiser un réseau d'amis avec Neo4J

```
$ MATCH (n:Personne) RETURN n LIMIT 25
```



Graphe

🕒 Exemple : matérialiser un réseau d'amis avec Neo4J

- Qui est ami avec DUPOND ?

- Premier niveau

```
$ match (p:Personne)-[:Est_Ami]->(amis) where p.nom='Dupond' return amis.nom
```

	amis.nom
Rows	Richard
	Durant

- Second niveau

```
$ match (p:Personne)-[:Est_Ami*2]->(amis) where p.nom='Dupond' return amis.nom
```

	amis.nom
Rows	Lajoie
	Roche

La subtilité est ici !

Graphe

🎯 Exemple : matérialiser un réseau d'amis avec Neo4J

- Quels sont tous les amis de Dupond ? (directs et indirects)

```
$ match (p:Personne)-[:Est_Ami*]->(amis) where p.nom='Dupond' return distinct amis.nom
```

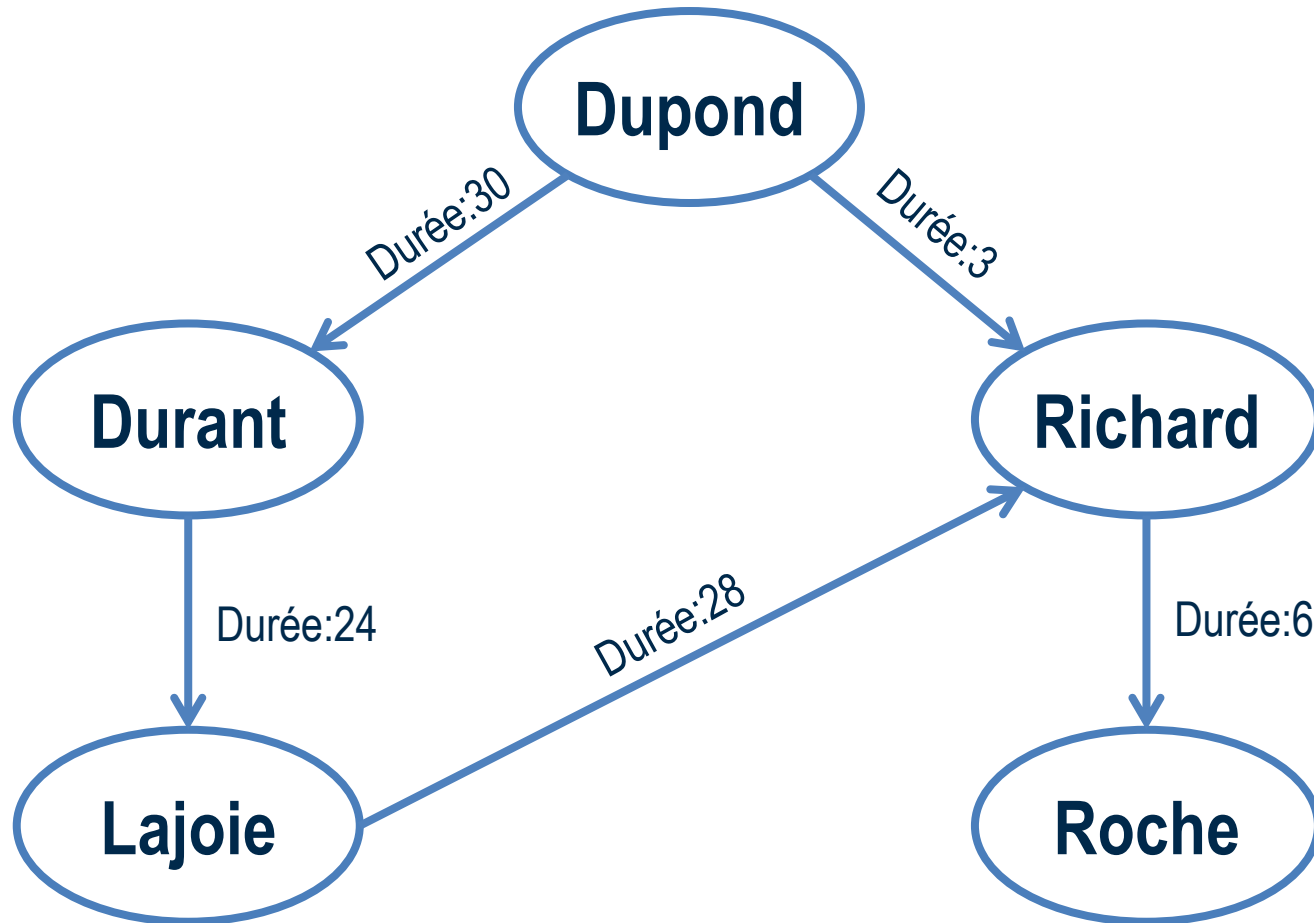
 amis.nom
Rows
Durant
Lajoie
Code
Richard
Roche

- Quelles sont les amitiés de plus de 10 ans

Graphe

🕒 Exemple : matérialiser un réseau d'amis avec Neo4J

- Quelles sont les amitiés de plus de 10 ans ?
 - On ajoute des propriétés aux relations



Graphe

🕒 Exemple : matérialiser un réseau d'amis avec Neo4J

○ Quelles sont les amitiés de plus de 10 ans ?

○ Et on demande

```
$ match (p1:Personne)-[r:Est_Ami]->(p2:Personne) where r.duree >=10 return r,p1,p2
```



	r	p1	p2
Graph			
Rows	duree 30	age 35 nom Dupond	nom Durant
Code	duree 24	nom Durant	nom Lajoie
	duree 28	nom Lajoie	nom Richard
Returned 3 rows in 1515 ms.			

Graphe

🎯 Cible

- Représentation de relations, de réseaux, d'organisations

🎯 Avantage

- Algorithmes de la théorie des graphes (chemin le plus court, degré de relation, ...)

🎯 Inconvénient

- Parcours complet de la base obligatoire pour avoir une réponse exhaustive

🎯 Implémentations

- Neo4J

Graphe

Implémentations

- ⦿ Neo4J
- ⦿ Titan
- ⦿ FlockDB
- ⦿ GraphBase
- ⦿ InfiniteGraph

Grappe

En production

⦿ Cisco

⦿ Ebay

Gestion des livraisons

⦿ SFR

⦿ TomTom

⦿ Lufthansa

⦿ Meetic

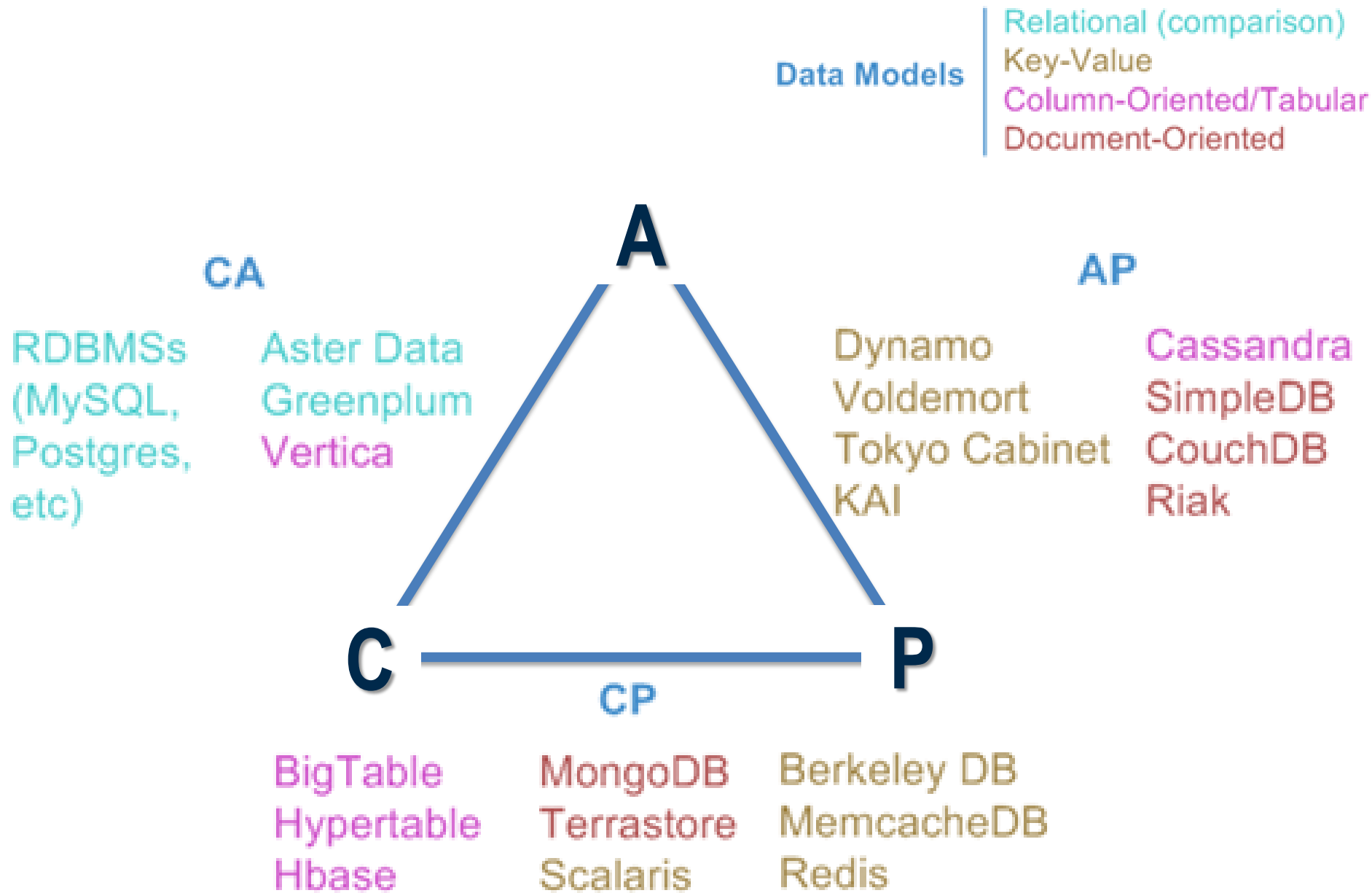


Les 4 grandes familles du NoSQL



- ⦿ Clés-valeurs
- ⦿ Document
- ⦿ Orientées colonnes
- ⦿ Graphes
- ⦿ **Le théorème CAP**

Le théorème CAP





Passons à la pratique



- ⦿ Clés-valeurs : Redis
- ⦿ Document : MongoDB
- ⦿ Orientées colonnes : Cassandra
- ⦿ Graphes : Neo4j

⦿ @ VM
192.168.56.4