

Séance 1

NoSQL vs SQL Historique et évolution



Ce(tte) œuvre est mise à disposition selon les termes de la Licence Creative Commons Attribution – Pas d'Utilisation Commerciale – Pas de Modification 4.0 International.

Objectifs

- Du relationnel au **NoSQL**
 - Comparaison des deux paradigmes
 - Historique de l'émergence du NoSQL
- Intérêt et utilisation du **NoSQL**
 - Big Data
 - Architecture distribuée
 - Aperçu des modèles de données

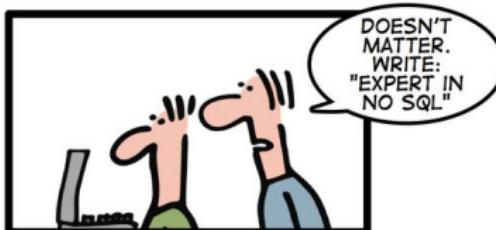
NoSQL

non relational

non SQL

not only SQL

HOW TO WRITE A CV



Leverage the NoSQL boom

Relationnel vs NoSQL



Informatique d'entreprise

- Beaucoup de changements dans les technologies utilisées
Langage de programmation, architecture, plateforme...
- Stabilité dans la manière de stocker les données
Utilisation de bases de données relationnelles
- Quelques challengers présents et fructueux dans des niches
Les architectes choisissent la base de données relationnelle

Logiciel et donnée

- Une entreprise utilise des **logiciels** et stocke des **données**
Ces deux éléments sont le plus indépendants possibles
- Les données **vivent souvent plus longtemps** que les logiciels
- Les données doivent être le **plus stable** possible

Compréhensibles facilement et accessibles par API

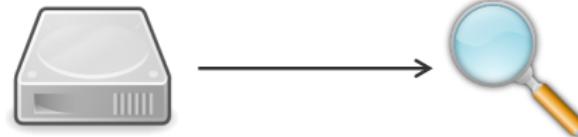
Donnée

- Besoin d'**organiser des données** au cœur de l'informatique

Optimiser la conservation et la restitution

- Plusieurs autres **sous-fonctions** importantes

Sécurité, protection contre les incohérences

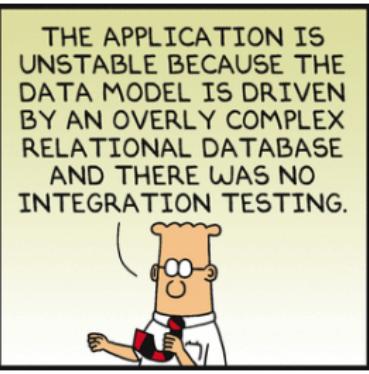


*Stockage sur
mémoire de masse*

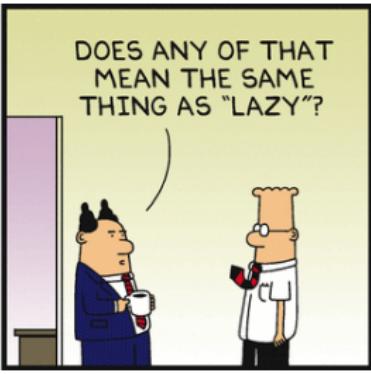
*Retrouver
les données*



Dilbert.com DilbertCartoonist@gmail.com



9-20-13 © 2013 Scott Adams, Inc. Dist. by Universal Uclick



I hate databases...

- La plupart des développeurs **n'aiment pas les bases de données**
 - Interaction avec un DBMS (*DataBase Management System*)
 - Apprentissage du langage SQL (*Structured Query Language*)
- Vu comme une intrusion d'un **élément externe**

Avec une très mauvaise intégration avec le code applicatif

Émergence du NoSQL

- Besoin de stocker de **grandes quantités** de données

Passage de grosses plateformes à des clusters de serveurs

- Bases de données relationnelles sous le nom de **NoSQL**

Cassandra, Mongo, Neo4j, Riak...

- Amoindrissement de la traditionnelle **consistance des données**

Pour la performance, montée en charge, programmation aisée

La force du relationnel

- Stockage efficace de données de manière **persistante**
Plus de flexibilité que le stockage dans des fichiers
- **Accès concurrent** aux données, en lecture et/ou écriture
Utilisation de transactions pour contrôler l'accès aux données
- **Intégration** et collaboration d'applications en entreprise
Réalisé avec une intégration par base de données partagée
- Modèle relationnel basé sur un **modèle standard**

La fin du relationnel ?

- Les bases de données relationnelles sont **puissantes et stables...**

Pas près de disparaître à court et moyen terme

- ...mais elles ne sont **plus suffisantes**

Lourdeur inutile pour le stockage de certaines données

- **Systèmes hybrides** combinant plusieurs technologies

De manière concurrente, coopérative, répartie, redondante...

Historique

- 1950 Développement du modèle hiérarchique (IMS)
- 1970 Apparition du modèle relationnel (Edgar F. T. Codd)
- 1980's Domination du modèle relationnel
- 2009 Invention du terme NoSQL

Modèle hiérarchique

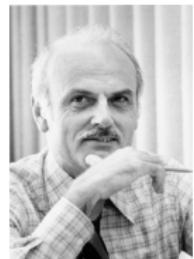
- Établissement de **relations** de parents vers des enfants
 - Relation unidirectionnelle*
- Base de données d'**enregistrements** avec des champs
 - Regroupés en types d'enregistrements*
- **Moteur IMS** (*Information Management System*) créé par IBM
 - Utilisé par la NASA pour gérer les matériaux de construction*

Modèle relationnel

- “A Relational Model of Data for Large Shared Data Banks”

Edgar Frank “Ted” Codd, Ph.D. (1923–2003)

IBM Research, San José, California, USA



- Turing Award 1981

- Organisation des données selon un **modèle mathématique**

Basé sur la théorie des ensembles et une algèbre relationnelle

- **Isolation de l'accès** aux données de l'implémentation physique

Grâce à un langage déclaratif de haut niveau

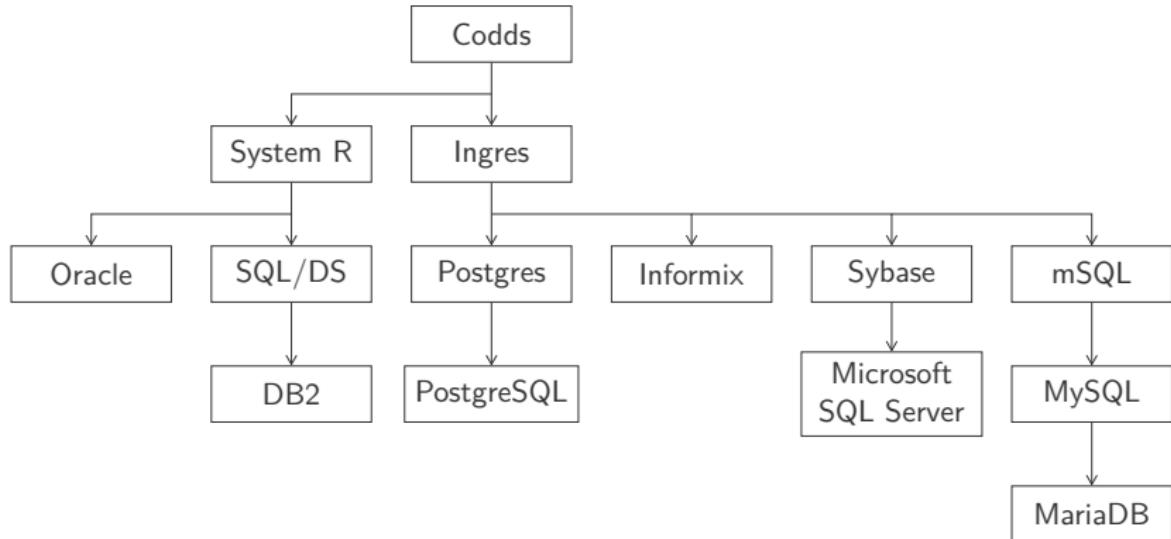
System R

- Première implémentation de SQL avec le prototype **System R**

Développé en 1974 pour expérimenter les concepts de Codd
- Langage **Sequel** (*Structured English Query Language*)
 - RSS (*Research Storage System*)
 - RDS (*Relational Data System*)
- **Pratt & Whitney** premier client de System R en 1977



Évolution du relationnel



De OLTP à OLAP

- **Online Transactional Processing (OLTP)**
Utilisation purement transactionnelle des données (gestion)
- **Online Analytical Processing (OLAP)**
Tableau de bord, analyse historique et prédictive (statistique)
- **Limitation** du modèle relationnel pour OLAP
Agrégats, optimisation de requête, indexation... pas suffisants

- Meetup par **Johan Oskarsson** au Hadoop summit @ SFO
Software developer basé à Londres
- **Choix d'un nom** court, mémorable, peu de résultat Google
#NoSQL “open-source, distributed, nonrelational databases”
- Plusieurs **caractéristiques** communes de ces bases de données
 - N'utilisent pas le modèle relationnel, ni SQL
 - Open source
 - Conçues pour être exécutées sur de large clusters
 - Basées sur les besoins des propriétés web du 21^e siècle
 - Pas de schéma, ajout de champs sans contrôle

Le monde du NoSQL



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

<http://nosql.mypopescu.com/kb/nosql>

Intérêt du NoSQL

- Housse de la productivité lors du développement
 - Gain de temps mapping de la base de données vers la mémoire
 - Moins de code à écrire, débugguer et faire évoluer
- Quantité de données à grande échelle
 - Stockage rapide de grandes quantités de données
 - Base de données répartie sur des clusters

Propriétés ACID

- Ensemble de **propriétés des transactions** en base de données
Atomicity, Consistency, Isolation et Durability
- Définition par **Reuter et Härder** en 1983
 - Une transaction fait tout ou rien (e.g. si panne de courant)
 - Base de données change d'un état valide vers un autre valide
 - Exécutions concurrentes de transactions comme séquentielles
 - Transaction validée confirmée et stockée

Propriétés BASE

- Gérer les **pertes de consistance** en maintenant la fiabilité

Basically Available, Soft state et Eventual consistency

- **Contraintes assouplies** par rapport aux propriétés ACID

- Toujours une réponse : failure, donnée inconsistante possible
- L'état change au cours du temps, même lorsque pas d'inputs
- Le système finira tôt ou tard par être consistant

Théorème CAP

- Théorème CAP, énoncé par Eric Brewer, sur le distribué
Consistency, Availability et Partition tolerance
- Au départ, distribution du calcul et maintenant des données
Clusters ou grids pour augmenter la puissance totale de calcul
- Trois garanties pas satisfaisables pour un système distribué
 - Consistance des données sur tous les nœuds
 - Disponibilité des données même avec perte d'un nœud
 - Résistance au morcellement avec nœuds autonomes

ACID ou BASE ? (1)

- ACID désiré dans un environnement “*shared something*”

Pessimiste : forcer la consistance et la fin des transactions

- A tout ou rien, commit ou rollback
- C pas de données inconsistentes
- I pas de connaissances des transactions concurrentes
- D transaction confirmée persistente

- BASE implémenté dans un environnement “*shared nothing*”

Optimiste : accepter des inconsistances temporaires

- BA garantie par réPLICATION
- S consistance à garantir par l'application
- E donnée périmée possible, consistance assurée tôt ou tard

ACID ou BASE ? (2)

- **Conjecture** liée au théorème de CAP

On ne peut satisfaire que deux des trois exigences CAP

- **Trois situations** possibles

- C + A ~ ACID : un seul serveur central (avec réPLICATION?)
- C + P : soit « w N, r 1 », ou « w 1, r N »(trop lent?)
- A + P = BASE : pas de consistance forte garantie

A perspective view of a tunnel formed by a repeating pattern of binary digits (0s and 1s). The tunnel is oriented vertically, with the pattern curving around it. The binary code is rendered in a light blue color against a dark background.

Big Data

Big Data

- Augmentation du **volume de données** manipulé

Notamment les entreprises et organismes liés à l'Internet

- **Multiplication exponentielle** jusqu'à des pétaoctets

- Données scientifiques, bases de données médicales
- Réseaux sociaux, opérateurs téléphoniques
- Indicateurs économiques et sociaux
- Agences nationales de défense du territoire

- **Défi** de gérer et traiter cette énorme quantité de données

Pas à la portée du traditionnel modèle relationnel

4Vs

- Big Data caractérisé par **quantité illimitée** de jeux de données
Données très complexes à collecter et à stocker
- Données suivent les **4Vs**
 - **Volume** de l'ordre de Pbytes, Ebytes
 - **Vélocité** taux de croissance des données et vitesse traitement
 - **Variété** avec plusieurs sources et types (image, vidéo, son...)
 - **Véracité** des données, obsolescence, intégrité et sécurité

Open Data

The screenshot shows the homepage of the Open NASA Data website. The header features the "open NASA" logo and a navigation bar with links for "Open Data", "Explore With Us", "Data Stories", "Innovation Space", and "About". The main section has a dark background with a starry space theme. A large white "Open Data" title is centered. Below it is a sub-headline: "We hope to spark your creative juices and equip you with data sets and tools to solve looming challenges here on Earth using NASA data, tools, and resources". To the right is a circular icon depicting a satellite in orbit around Earth. The lower half of the page is white and contains three sections: "Data", "Code", and "APIs", each with an image and a brief description.

Open Data

We hope to spark your creative juices and equip you with data sets and tools to solve looming challenges here on Earth using NASA data, tools, and resources

We're Setting Data, Code and APIs free

Data

This inventory offers services that enable users the ability to discover, collaborate,

Code

A community-driven catalog of NASA's Software Release projects developers can

APIs

Developers can register for API keys and look up available API's, hooks for analytics,

Open Data

open NASA

Open Data

We hope to spark your creative tools to solve looming challenges and resources

We're Setting Data, Data

This inventory offers services that enable users the ability to discover, collaborate,

Open Data Explore With Us Data Stories Innovation Space About

OPEN GOVERNMENT DATA U.S. DEPARTMENT OF DEFENSE

Search Open Government Data

HOME OPEN DATA FEATURED DATASET FEATURED API PUBLIC DATA LISTING CONTACT US

Featured API

Each month, we feature one application programming interface (API) provided by the Department of Defense. These APIs allow developers to produce new and innovative applications using open government data.

[View](#)

■ ■ ■

Links

[PUBLIC DATA LISTING](#)

[DOD OPEN GOV](#)

[DoD Developer Page](#)

[DoD Mobile Applications Gallery](#)

[DoD Digital Strategy Page](#)

Open Government Data at the Department of Defense

The Department of Defense (DoD) maintains a vast amount of information. Information that most folks don't normally associate with Defense. Consider the following:

GPS

The Global Positioning System (GPS) is a space-based satellite navigation system built and maintained by DoD and is freely available to anyone in the world with a GPS receiver. In addition to navigation, uses of GPS include precise timing for financial transactions, search and rescue, communications, farming, recreation and both military and commercial aviation. GPS is operated by the **2nd Space Operations Squadron** at Schriever Air Force Base, Colorado.

Open Data

open NASA

Open Data Explore With Us Data Stories Innovation Space About

Sitemap | Legal notice | Contact English (en) ▾

European Union Open Data Portal

EUROPA > Open Data Portal > Data

Data Applications Linked Data Developers' corner About

Find datasets... Share

Show results with: all of these words any of these words the exact phrase

Total datasets available: 9070

Most viewed datasets

All view all ↗

- DGT-Translation Memory (14773 views)
- [LATEST VERSION] Elevation map of Europe (8999 views)
- Tenders Electronic Daily (TED) – public procurement notices from the EU and beyond (8021 views)
- CORDIS – EU research projects under FP7 (2007–2013) (7902 views)
- EuroVoc, the EU's multilingual thesaurus (7090 views)
- Transparency Register (5292 views)
- Register of Commission documents (7032 views)

Browse datasets by subject or groups

Employment and working conditions Social questions Economics Finance

Trade Industry Education and communications Science

more subjects ↗

Popular terms

balance of payments

The Global Positioning System (GPS) is a space-based satellite navigation system built and maintained by DoD and is freely available to anyone in the world with a GPS receiver. In addition to navigation, uses of GPS include precise timing for financial transactions, search and rescue, communications, farming, recreation and both military and commercial aviation. GPS is operated by the 2nd Space Operations Squadron at Schriever Air Force Base, Colorado.

EN GOV

Open Page

Applications

Strategy

Open Data

The screenshot shows the homepage of Data.gov.be. At the top, there are language links (NL, EN, FR, DE) and a logo for open NASA. On the right, there's a link to other official information and services (www.belgium.be) and a 'be' icon. The main heading is 'Data, tools and resources. More than 5000 datasets.' Below this is a search bar and a grid of icons representing different sectors: Agriculture and Fisheries, Culture and Sports, Economy and Finance, Education, Energy, Environment, Health, International, Justice, Population, Public Sector, Regional, Science and Technology, and Transport. At the bottom, there are links for Terms of Use, GitHub, and Your app?. A specific dataset titled 'balance of payments' is highlighted, showing 5292 views and a link to Register of Commission documents.

NL EN FR DE
open NASA www.belgium.be be

Data, tools and resources. More than 5000 datasets.

Search

Agriculture and Fisheries Culture and Sports Economy and Finance Education Energy Environment Health

International Justice Population Public Sector Regional Science and Technology Transport

Terms of Use GitHub Your app ?

(5292 views)
[Register of Commission documents](#)

balance of payments

The Global Positioning System (GPS) is a space-based satellite navigation system built and maintained by DoD and is freely available to anyone in the world with a GPS receiver. In addition to navigation, uses of GPS include precise timing for financial transactions, search and rescue, communications, farming, recreation and both military and commercial aviation. GPS is operated by the **2nd Space Operations Squadron** at Schriever Air Force Base, Colorado.

Principaux acteurs



Évolution technologique

- Premier SGBD construits autour de **mainframes**

Avec les limitations des capacités de stockage de l'époque

- Plusieurs **évolutions technologiques** ont levés ces contraintes
 - Généralisation des interconnexions de réseau
 - Augmentation de la bande passante sur Internet
 - Diminution du cout des machines moyennement puissantes

Google FS

- Système de fichiers distribué propriétaire développé par Google
Google File System (GFS) présenté en 2003
- Stockage redondant et résilient sur un **cluster de machines**
Puissance moyenne et « jetables » (commodity hardware)
- Plusieurs **caractéristiques** de ce système de fichiers
 - Conçu pour des interactions système-système
 - Exécuté dans l'espace utilisateur et pas dans le kernel de l'OS
 - Gère des fichiers de plusieurs gigaoctets
 - RéPLICATION automatique des données par les chunkservers

MapReduce

- Modèle de programmation et implémentation associée
 - Traitement et génération des grandes quantités de données
 - Algorithme parallèle et distribué sur un cluster
- Implémentation basée sur deux fonctions
 - Map effectue un traitement sur une liste (tri, filtre...)
 - Reduce regroupe les données en un résultat (somme...)

Apache Hadoop

- Hadoop est une implémentation libre de MapReduce en Java

Par Doug Cutting, nommé comme l'éléphant doudou de son fils

- Hadoop Distributed FileSystem (HDFS)

Inspiré par la publication sur GFS

- Utilisé par de nombreuses entreprises

- Soutien de Microsoft, utilisation sur Windows Azure et Server
- Cluster Yahoo ! de 4000 machines, bientôt 10000 avec la 2.0
- Facebook annonce l'installation de HDFS avec 100 pétaoctets



BigTable

- Système de **gestion de données** basé sur GFS
Gigantesque table de hachage distribuée
- Gestion de la **cohérence** des données et **distribution** sur GFS
- Plusieurs implémentations libres dont **HBase** (Apache)
Utilisé notamment par eBay, Yahoo ! et Twitter

Dynamo

- Dépôt de **paires clé-valeur** distribué et propriétaire (Amazon)

Mise en œuvre par Amazon dans Simple Storage Service (S3)

- **Quatre principes** clés

- Évolutivité incrémentale sans influence sur opérateur et système
- Symétrie avec tous les nœuds étant égaux
- Décentralisation avec aucun contrôle central
- Hétérogénéité en partageant le travail selon les ressources

- Création de plusieurs **moteurs NoSQL** basés sur Dynamo

Cassandra, Riak, Projet Voldemort (LinkedIn)

Modèle de données



Défaut d'impédance (1)

- Défaut d'impédance objet-relationnel avec SQL

Passer du relationnel à l'objet s'effectue avec une impédance

- Différence entre modèle relationnel et structures en mémoire

Relations et tuples versus structures de données complexes

- Apparition des langages de programmation orienté objet

Object-Relational Mapping (ORM) comme Hibernate...

Défaut d'impédance (2)

```
1 class Address:  
2     def __init__(self, street, number, zipcode, city):  
3         self.__address = (street, number, zipcode, city)  
4  
5     def __str__(self):  
6         return '{}, {}{}\n{} {}'.format(*self.__address)  
7  
8 ecam = Address("Promenade de l'Alma", 50,  
9                 1200, "Woluwé-Saint-Lambert")
```

Address				
Id	Street	Number	CityID	
1	Promenade de l'Alma	50	1	

City		
Id	Zipcode	Name
1	1200	Woluwé-Saint-Lambert

Intégration versus Application (1)

- Coordination de plusieurs applications **autour des données**
Partage des données dans une seule base de données commune
- **Difficile de changer** la structure de la base de données
Non trivial d'assurer l'intégrité des données



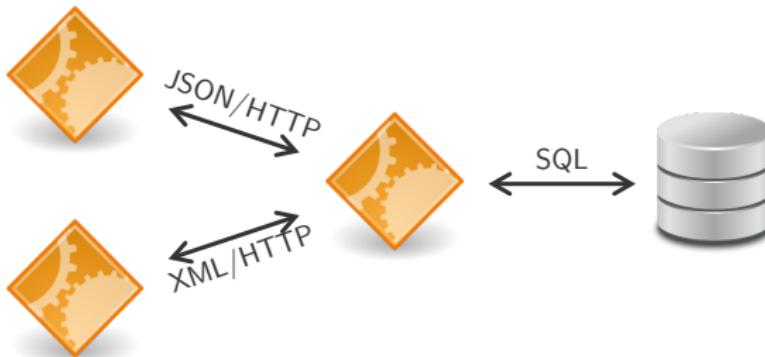
Intégration versus Application (2)

- Accès base de données par **une seule application**

Offre une interface d'accès aux autres applications

- Déploiement de services web et **architecture orientée services**

Plus grande flexibilité dans le format des données échangées



Modèle de données (1)

- Modèle avec lequel on perçoit et manipule les données

Différent du modèle de stockage sur disque

- Modèle relationnel constitué de tables avec des rangées

Colonnes avec des valeurs pouvant référencer d'autres rangées

- Changement vers une collection d'agrégats

Unité d'information traitée, stockée, échangée de façon atomique

Modèle de données (2)

- Quatre principaux **modèles de données** pour le NoSQL

Analyse détaillée d'un exemple par modèle

- **Pas de classification** unique et non ambiguë

Certaines bases de données couvrent plusieurs modèles

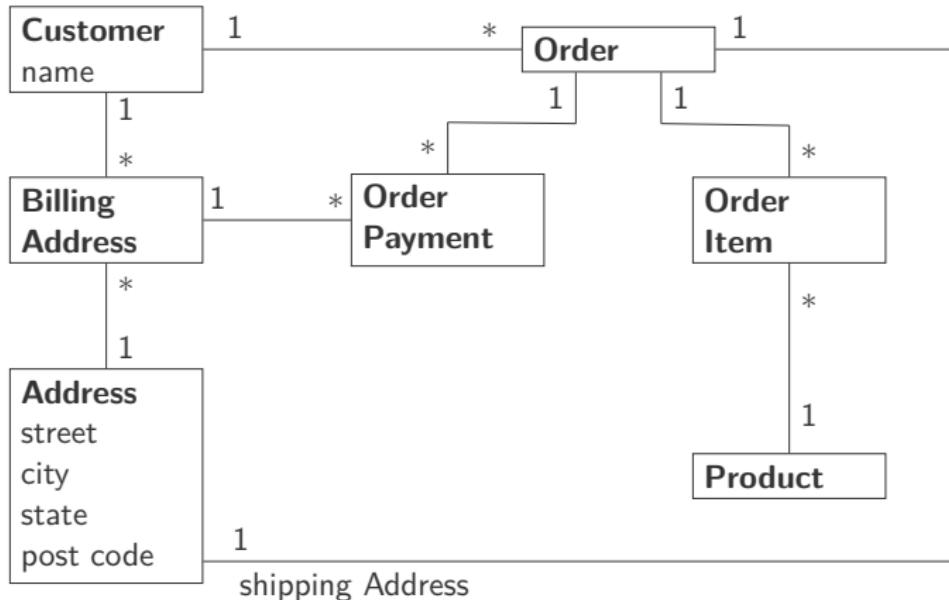
Modèle de données	Exemples de base de données
Clé-valeur	BerkeleyDB, Memcached, Redis , Riak...
Document	CouchDB, MongoDB , OrientDB...
Colonne	Amazon Simple DB, Cassandra , HBase...
Graphe	FlockDB, HyperGraphDB, Neo4j , OrientDB...

Agrégat

- Opération sur des **unités de données** complexes et structurées
Pour dépasser les limitations des tuples du modèle relationnel
- Possibilité d'**imbriquer** des listes et autres structures
Traitement des différents “objets” comme des unités
- Unité de manipulation et de gestion de la **concurrence**
Facilitation de la répartition des données sur des clusters

Relation vs agrégat (1)

- Modèle complètement **normalisé** sans données dupliquées



Relation vs agrégat (2)

Customer	
Id	Name
1	Martin

Order		
Id	CustomerId	ShippingAddressId
99	1	77

Product	
Id	Name
27	NoSQL Distilled

BillingAddress		
Id	CustomerId	AddressId
55	1	77

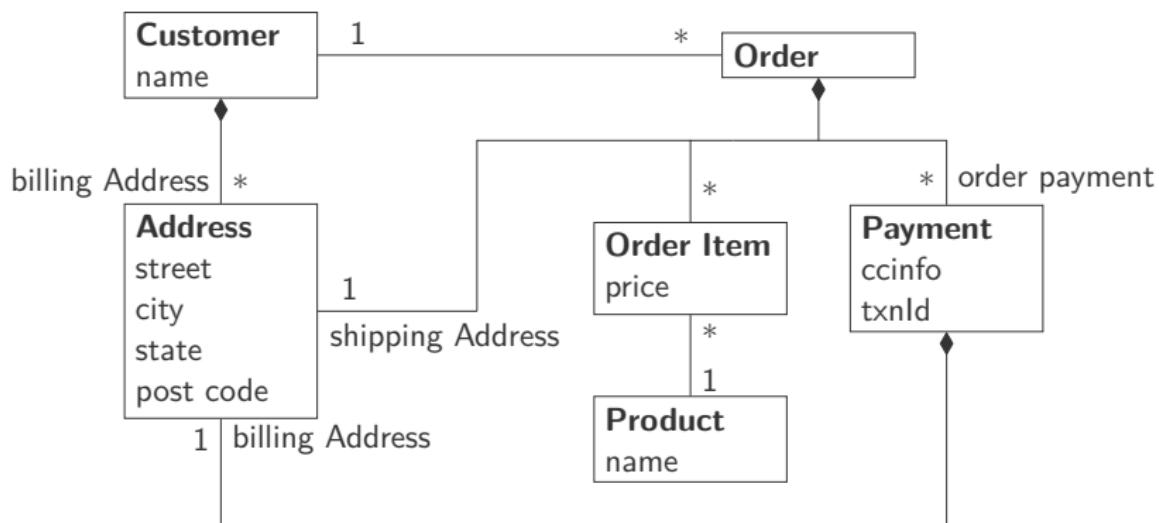
OrderItem			
Id	OrderId	ProductId	Price
100	99	27	32.45

Address	
Id	City
77	Chicago

OrderPayment				
Id	OrderId	CardNumber	BillingAddressId	txnId
33	99	1000-1000	55	abelif879rft

Relation vs agrégat (3)

- Modèle composé de **deux agrégats** principaux



Relation vs agrégat (4)

```
1 # Customer
2 {
3     "id": 1,
4     "name": "Martin",
5     "billingAddress": [{"city": "Chicago"}]
6 }
7
8 # Order
9 {
10    "id": 99,
11    "customerId": 1,
12    "orderItems": [
13        {"productId": 27,
14         "price": 32.45,
15         "productName": "NoSQL Distilled"
16     },
17     "shippingAddress": [{"city": "Chicago"}],
18     "orderPayment": [
19         {"ccinfo": "1000-1000",
20          "txnid": "abelif879rft",
21          "billingAddress": {"city": "Chicago"}
22     ]
23 }
```

Clé-valeur

- Stockage d'agrégats sous la forme **clé-valeur**

La clé joue le rôle d'identifiant unique de chaque agrégat

- Récupération d'un agrégat **à partir de sa clé**

Fonctionne comme une table de correspondance



Document

- Stockage d'agrégat sous la forme de **document**
Chaque document est identifié de manière unique par un ID
- Récupération du document ou d'une **partie de document**
À partir de requêtes sur les champs de l'agrégat
- Création d'**index** sur base du contenu des documents
Pour accélérer les opérations de recherche



Colonne

- Stockage sur disque **des colonnes** au lieu des lignes

On peut voir le stockage comme un map à deux niveaux

- Structure clé-valeur avec **identifiant de ligne** comme clé

Le second niveau contient les informations sur les colonnes



Graphe

- Possibilité de **relation** entre les agrégats
Avec possibilité de mise à jour automatique
- Utile pour des petits enregistrements avec **beaucoup de liens**
Ensemble de nœuds connectés par des arêtes
- Réseaux sociaux, préférences, règles d'admissibilité...
“Quelles sont toutes les choses qu'aiment Alexis et Sébastien ?”





Caractéristiques du NoSQL

Base de données sans schéma

- Les bases de données NoSQL n'ont pas de **schéma de données**

Contrairement à la structure rigide imposée par le relationnel

- **Ajout libre** de données de n'importe quel type

Comme clé, document, colonne, arête et propriétés

- Autorise le stockage de **données non uniformes**

Ce qui élimine le besoin d'avoir des valeurs NULL

Schéma implicite

- Hypothèses sur la structure des données dans le code
La base de données reste ignorante, c'est l'application qui vérifie
- Danger si plusieurs applications sur la même base de données
Elles doivent se mettre d'accord sur le schéma des données
- La migration de données doit toujours se faire prudemment
Que ce soit en relationnel ou en NoSQL

Développeur au centre

- Méthodologie de développement centrée sur le **développeur**
 - Design et mise en œuvre de l'architecture de l'application
 - Modélisation des données
- Deux approches différentes **RDBMS vs NoSQL**
 - Modèles de données relationnels théoriques
 - Requêtes et configuration applicative à supporter

Architecture du NoSQL

- Construction de la DB sur le **relationnel** avec DBMS
 - Description structure de données et du stockage
 - Processus de récupération des données et fiabilité
 - Données en tables (enregistrement et colonne) et non répétées
 - Importance des clés primaires
- Gestion des données beaucoup **plus flexible** en NoSQL
 - Répartition sur plusieurs serveurs, plateformes, processeurs
 - Évolution graduelle du schéma (implicite) des données

Opérations et relations sur les données

- Opérations classiques de **type CRUD** avec les RDBMS
Create, Read, Update, Delete
- Opérations beaucoup plus **diverses et variées** en NoSQL
 - Grand nombre d'ajouts et de mises à jour
 - Opérations sur d'autres entités que des lignes de tables
- NoSQL pas adapté pour données avec **beaucoup de relations**
RDBMS ont des one-to-one, one-to-many, many-to-many

Crédits

- Photos des livres depuis Amazon
- Photos des logos depuis Wikipédia
- <http://geekandpoke.typepad.com/geekandpoke/2011/01/nosql.html>
- <https://www.flickr.com/photos/15216811@N06/13495437084>
- <https://openclipart.org/detail/34537/tango-drive-hard-disk>
- <https://openclipart.org/detail/59167/magnifying-glass>
- <http://dilbert.comstrip/2013-09-20>
- https://en.wikipedia.org/wiki/File:Edgar_F_Codd.jpg
- <https://www.flickr.com/photos/75279887@N05/6914441342>
- <https://www.flickr.com/photos/lukevu/6380430175>
- <https://openclipart.org/detail/94723/database-symbol>
- <https://openclipart.org/detail/35407/tango-applications-other>
- <https://www.flickr.com/photos/diegolizcano/35522041701>