Big Data Management

Master 2 MALIA

Année 2023-2024

Jérôme Darmont

https://eric.univ-lyon2.fr/jdarmont/

Actualités du cours

https://eric.univ-lyon2.fr/jdarmont/?page_id=3144

https://eric.univ-lyon2.fr/jdarmont/?feed=rss2

https://social.sciences.re/@darmont #maliabdm

Partie 1

Lacs de données

www.smartdatacollective.com

Plan (partie 1)

- Définitions
- Entrepôts et lacs de données
- Métadonnées et modèles de métadonnées
- Architectures et technologies pour les lacs
- Discussion, travaux de recherche

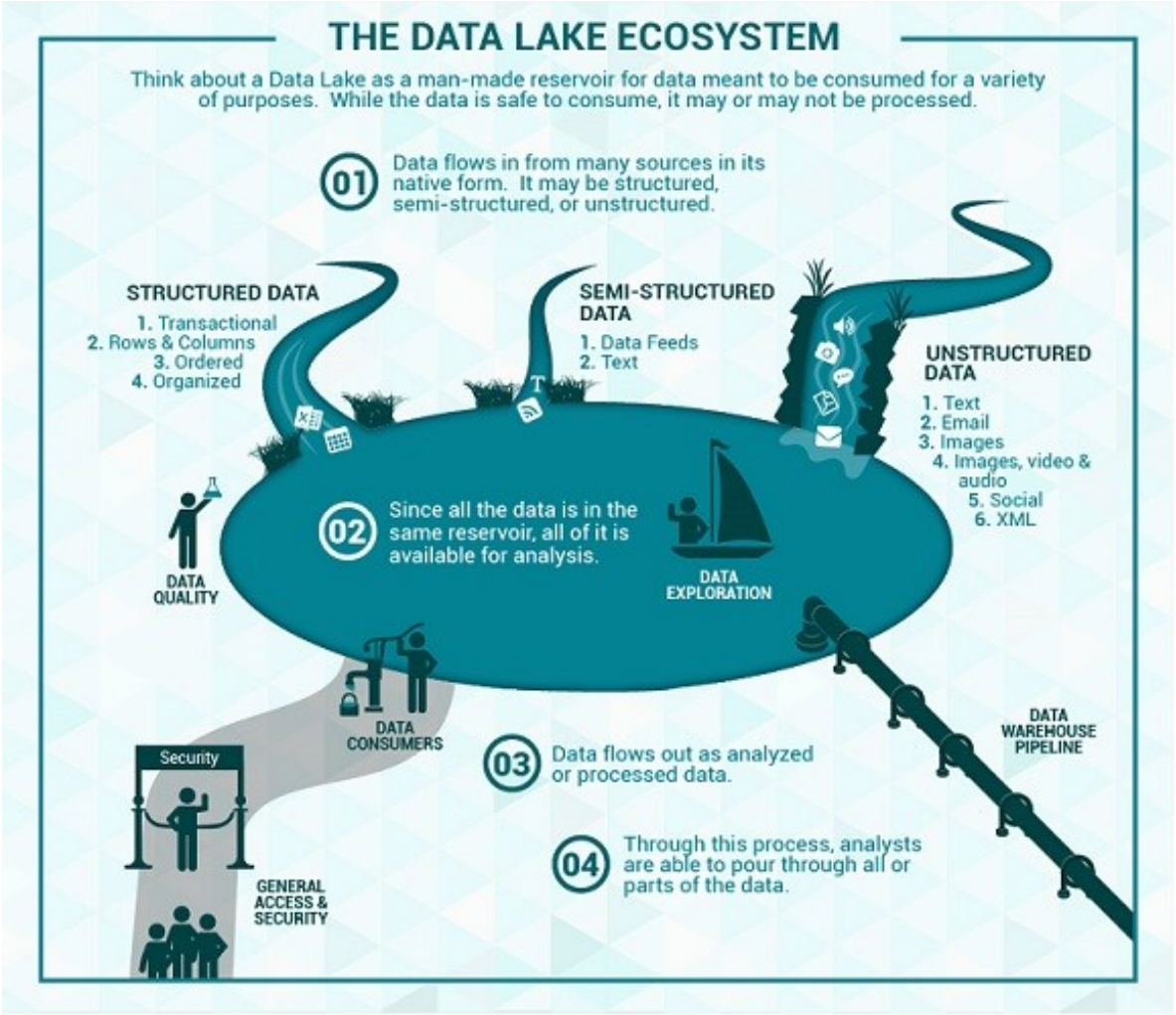
Définition de James Dixon (2010)

“If you think of a datamart as a store of bottled water – cleansed and packaged and structured for easy consumption – the data lake is a large body of water in a more natural state.

The contents of the data lake stream in from a source to fill the lake, and various users of the lake come to examine, dive in, or take samples.”

csgsolutions.com

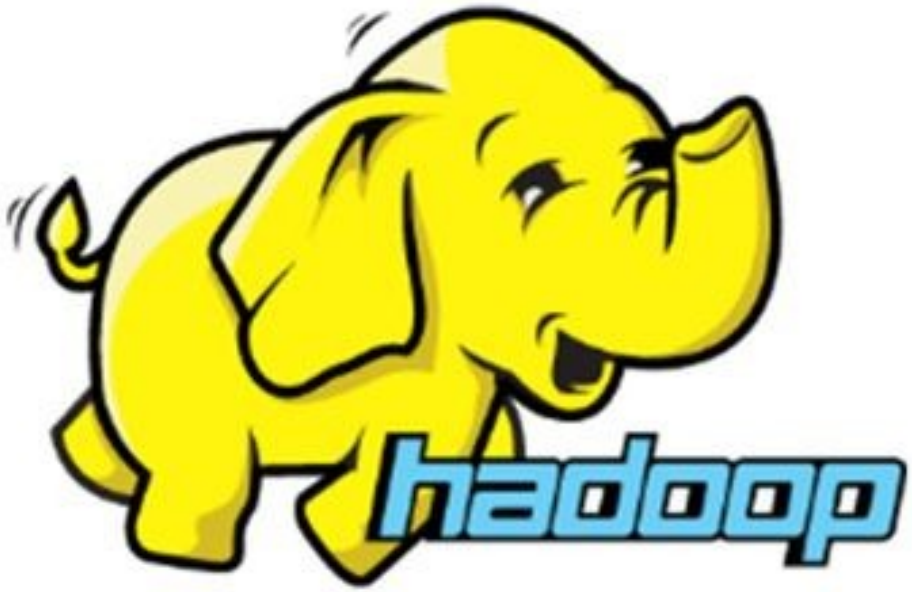
Plus en détail



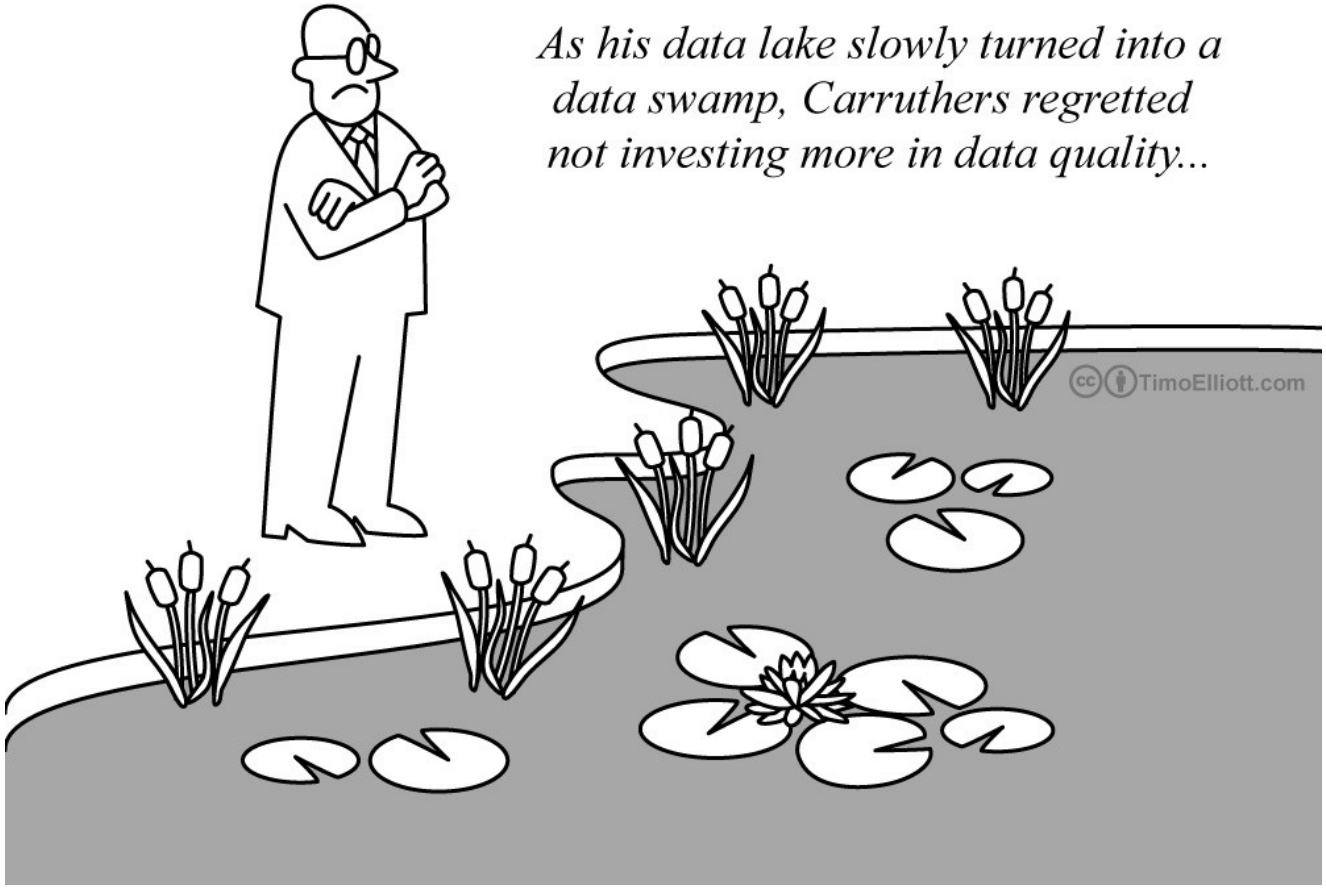
dunnsolutions.com

Définition initiale de l'industrie

…et de quelques auteurs académiques (minoritaire aujourd'hui)



Danger 1 : le *data swamp*

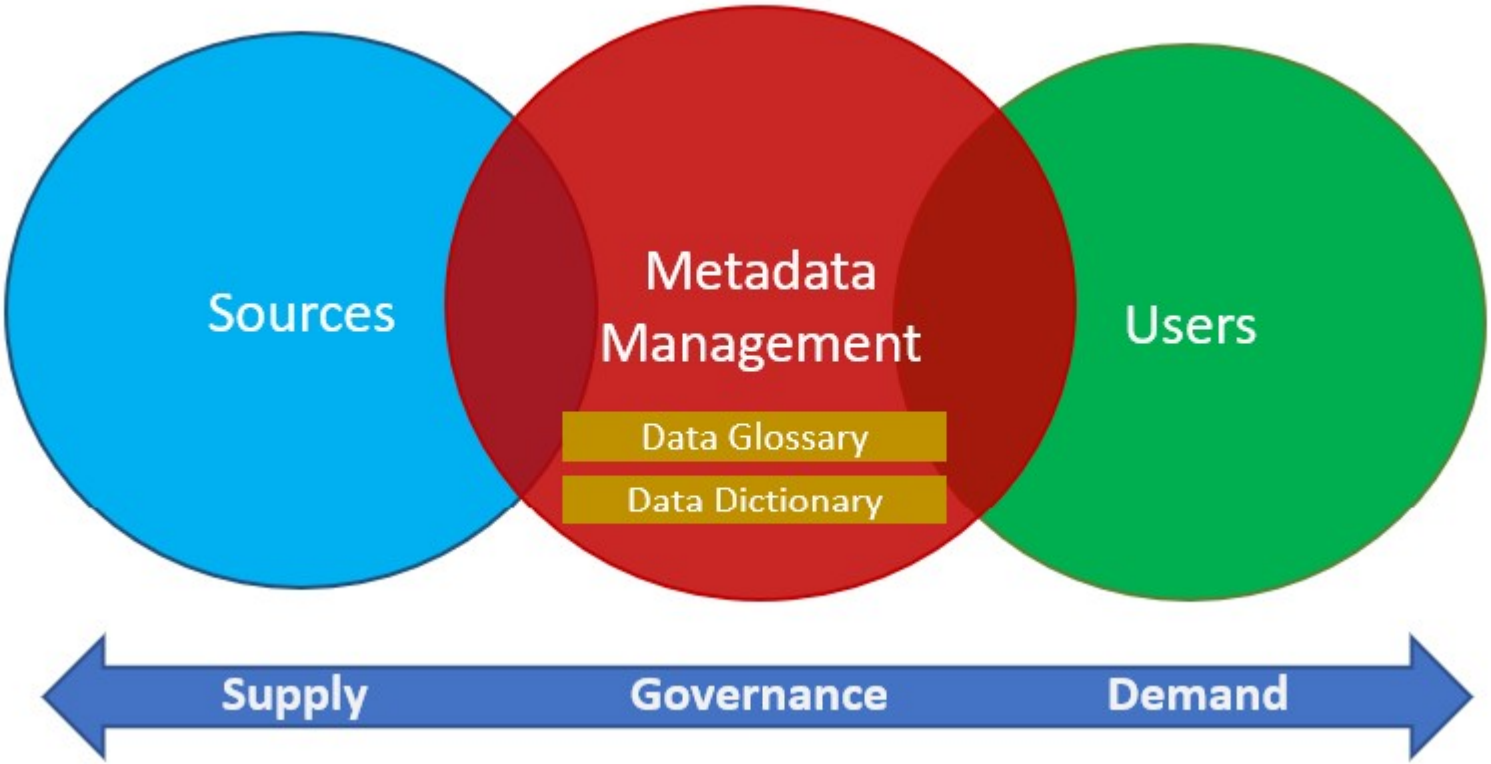


timoelliott.com

Danger 2 : le *data dump*

www.cartoonstock.com

Comment éviter ces dangers ?



medium.com

+ cycle de vie des (méta)données

Définition de Scholly et al. (2019)

Lac de données

Système **évolutif** (passage à l'échelle) de stockage et d'analyse

Données de tous types dans leur **format natif**

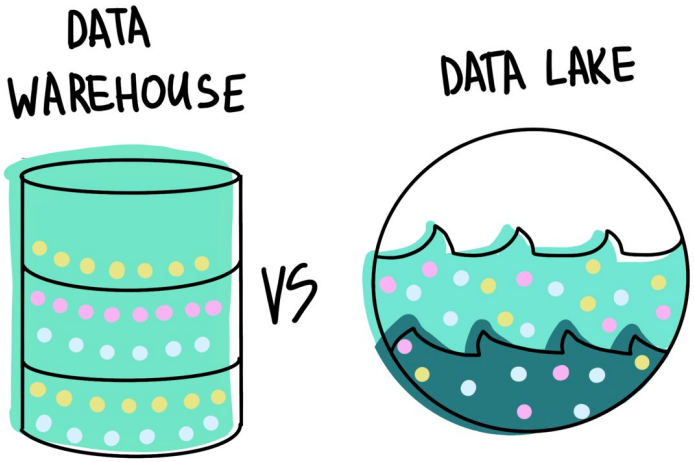
Utilisé **pas seulement** par des *data scientists* et *data analysts*

(utilisateur·trices métiers, chercheur·es...)

Définition de Scholly et al. (2019)

Caractéristiques des lacs de données
Catalogue de métadonnées (qualité des données)
Politique et outils de **gouvernance des données**
Ouverture à **tous types d'utilisateur·trices**
Intégration de **tous types de données**
Organisation **conceptuelle, logique et physique**
Passage à l'échelle (stockage et traitement)

Plan (partie 1)

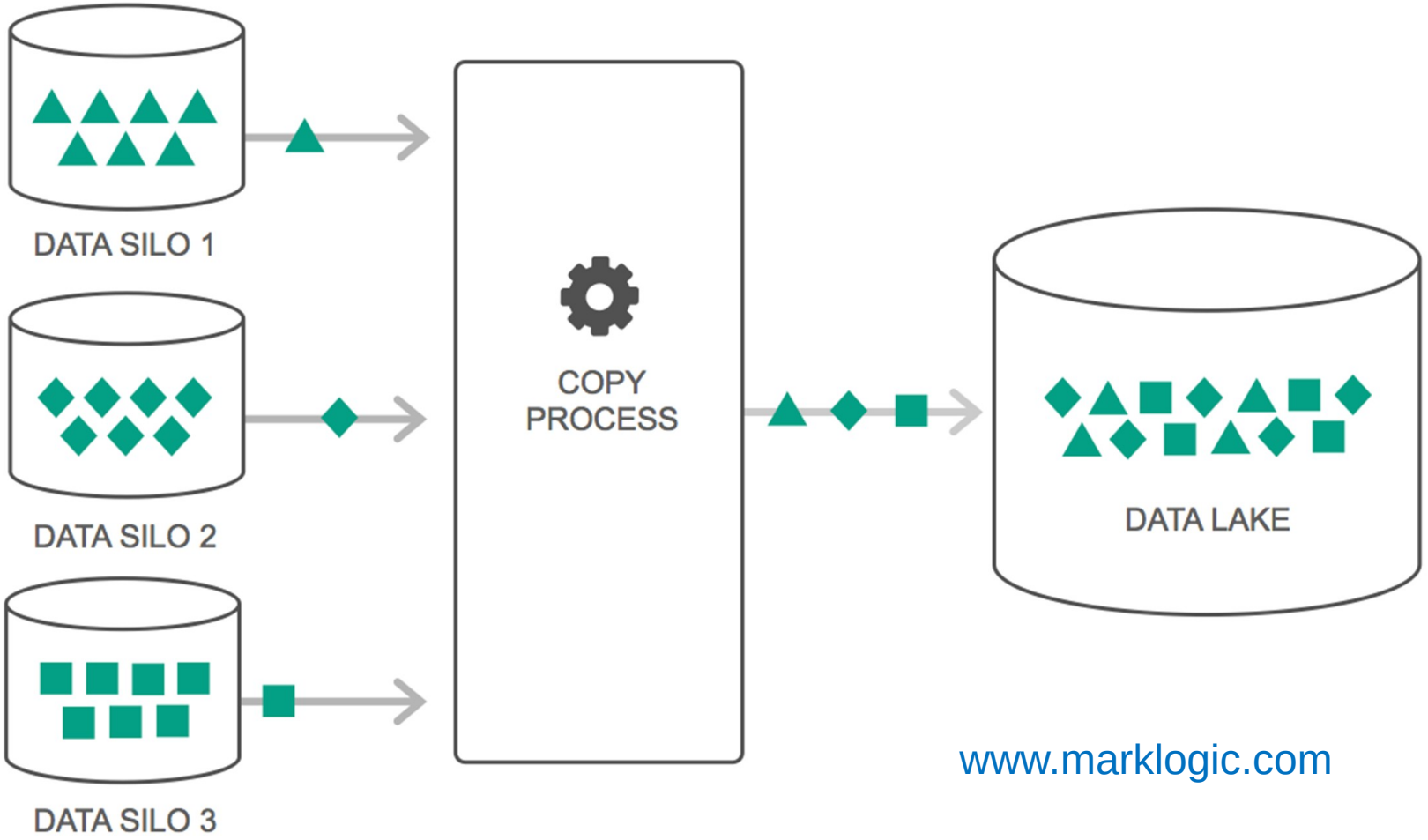


- ✓ Définitions
- Entrepôts et lacs de données
- Métadonnées et modèles de métadonnées
- Architectures et technologies pour les lacs
- Discussion, travaux de recherche

Data lake *vs.* Data warehouse

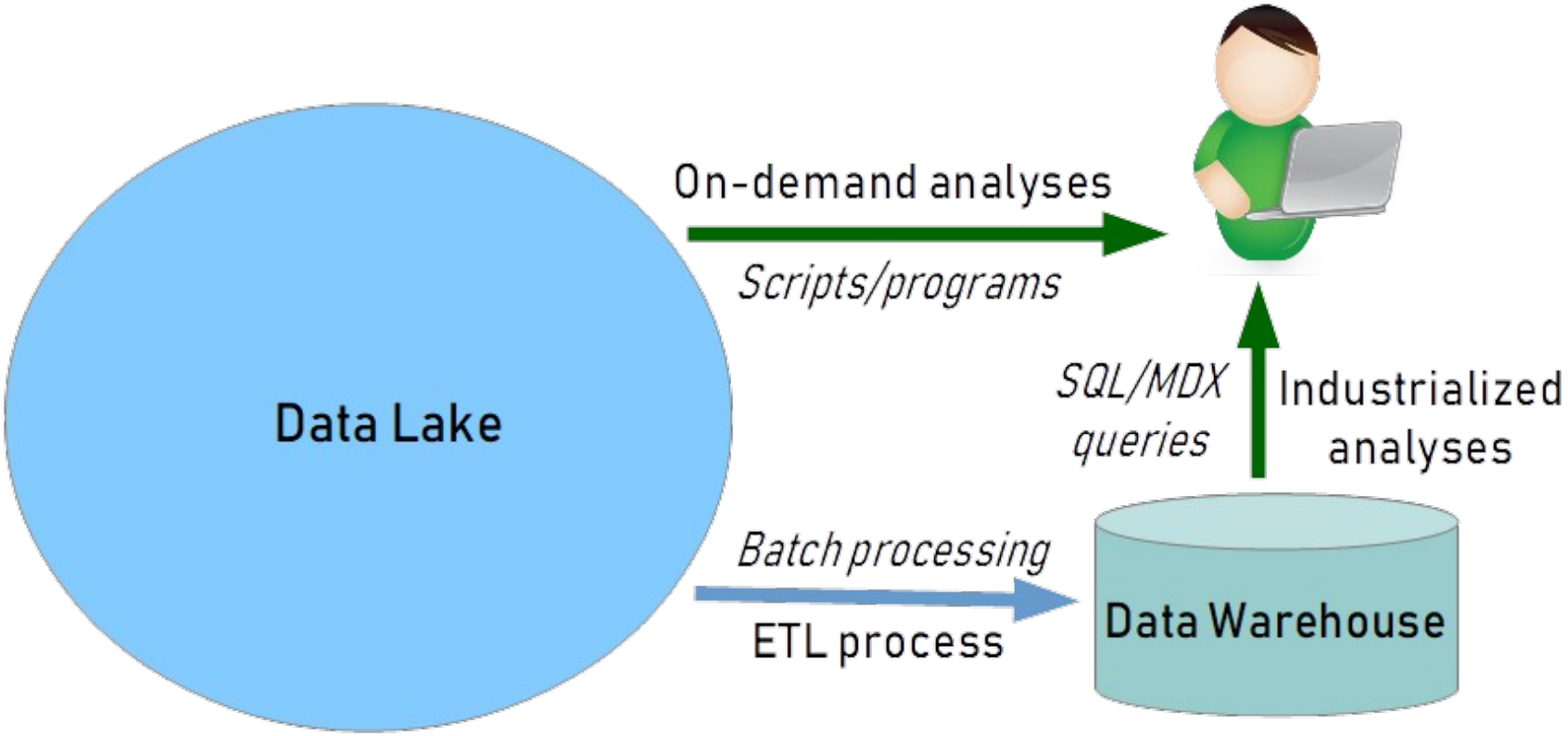
Domaine	Entrepôt de données	Lac de données
Données	Nettoyées	Brutes
Schéma	*Schema on write*	*Schema on read*
Sources de données	En nombre limité	Multiples
Structure	Données structurées	Données structurées, semi-structurées, non-structurées
Analyses	Industrialisées	À la demande
Accès aux données	SQL/MDX	Scripts/programmes
Coût de stockage	Peu coûteux	Très peu coûteux
Intégration données	ETL	ELT
Perte d'information	Agrégation	Aucune
Architecture	Figée, silos	Flexible
Utilisateurs	Experts métier	*Data scientists*

Entrepôt comme source d'un lac



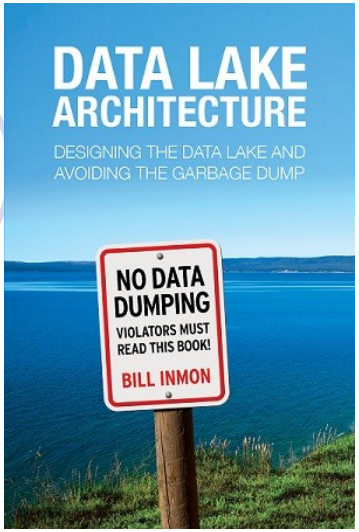
www.marklogic.com

Lac comme source d'un entrepôt



Sawadogo et Darmont 2019

Entrepôt dans un lac

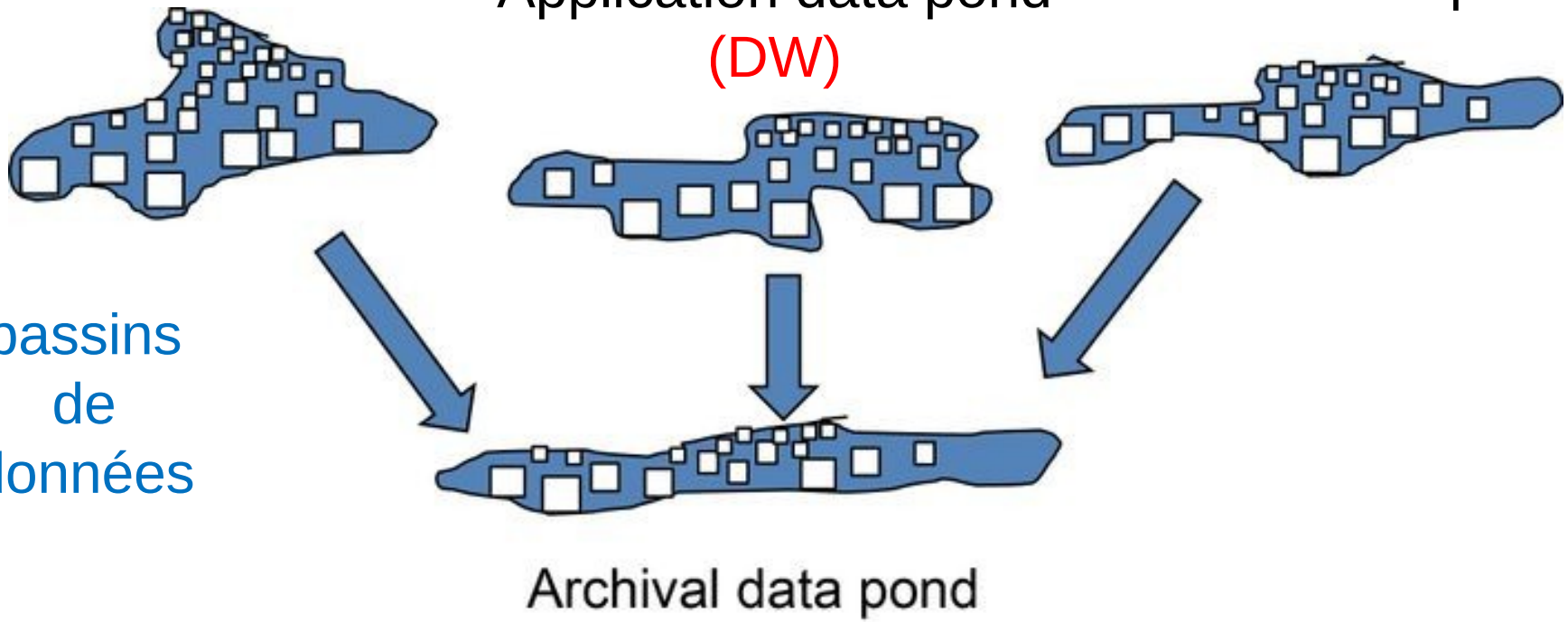


(Raw data pond)

Analog data pond

Application data pond
(DW)

Textual data pond



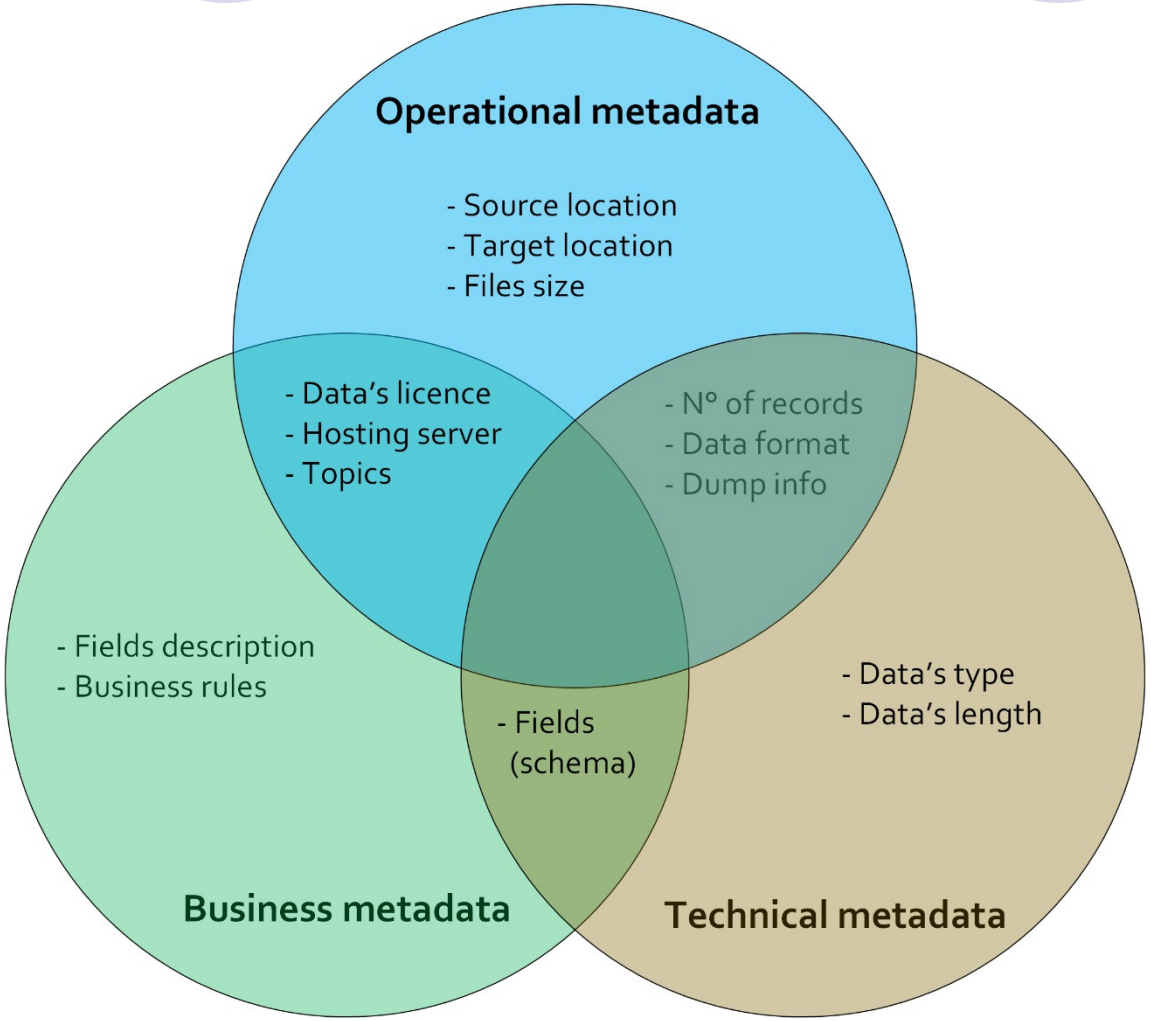
bassins
de
données

Archival data pond

Plan (partie 1)

- ✓ Définitions
- ✓ Entrepôts et lacs de données
- Métadonnées et modèles de métadonnées
- Architectures et technologies pour les lacs
- Discussion, travaux de recherche

Typologie de Diamantini et al. (2018)



Typologie de Sawadogo et Darmont (2019)

Généralisation de Maccioni and Torlone (2018)

Objet = ensemble de données homogènes
Table relationnelle, fichiers (XML, tableur, texte, multimédia…)

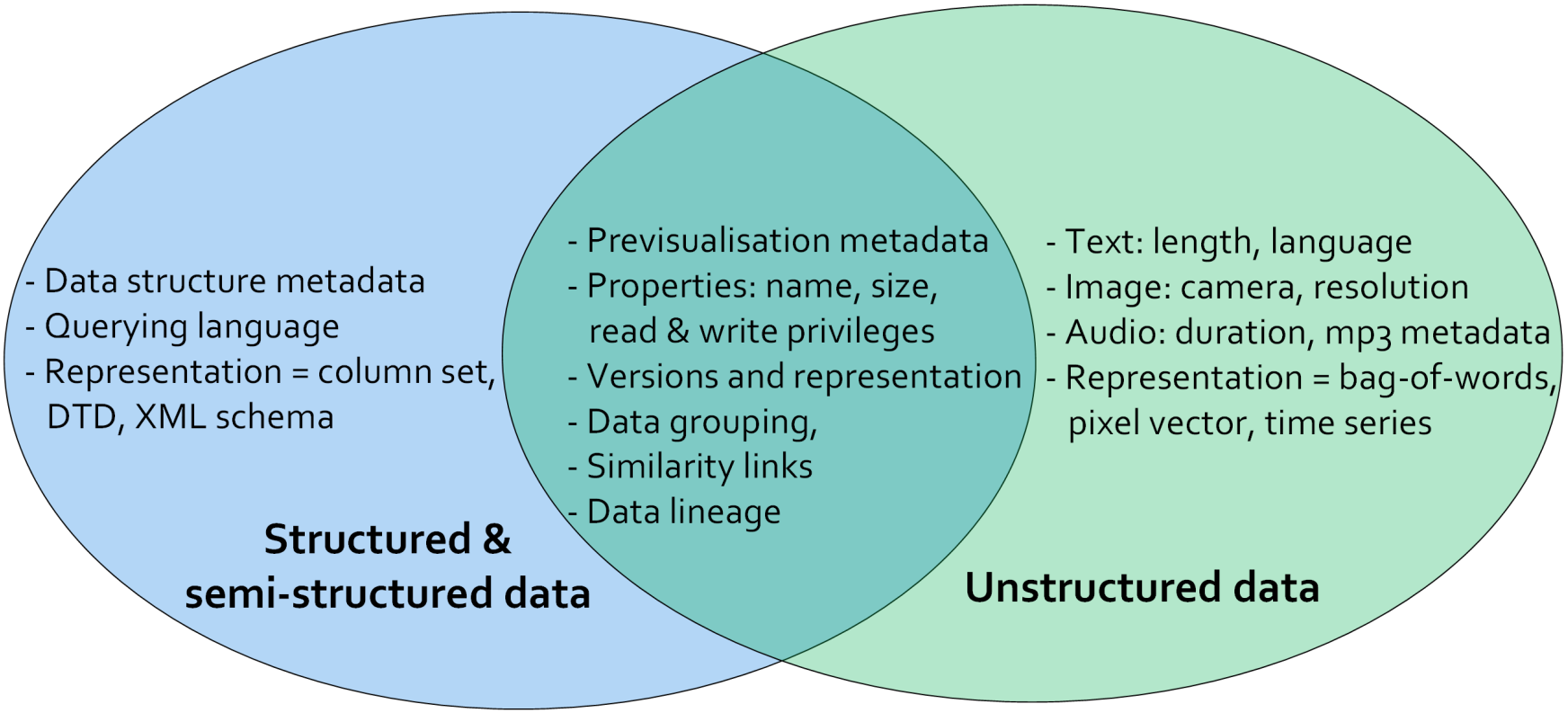
Métadonnées intra-objet	Exemples
Propriétés	Nom de fichier, taille, date de création…
Prévisualisations/résumés	Schéma, nuage de mots...
Versions et représentations	Transformation des données
Métadonnées sémantiques	Description, catégorie...

Typologie de Sawadogo et Darmont (2019)

Métadonnées inter-objets	Exemples
Regroupements	Thématiques, par langue...
Similarités	Via des mesures de similarité
Parentés	Jointures, unions...

Métadonnées globales	Exemples
Ressources sémantiques	Ontologies, taxonomies...
Index	Index inversés
Journaux	*Logs*

Typologie opérationnelle

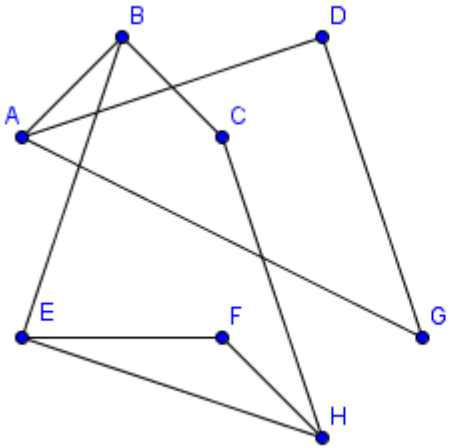


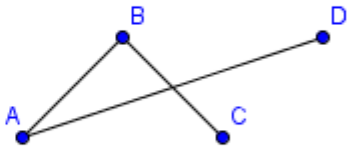
Sawadogo et Darmont 2019

Représentation logique des métadonnées

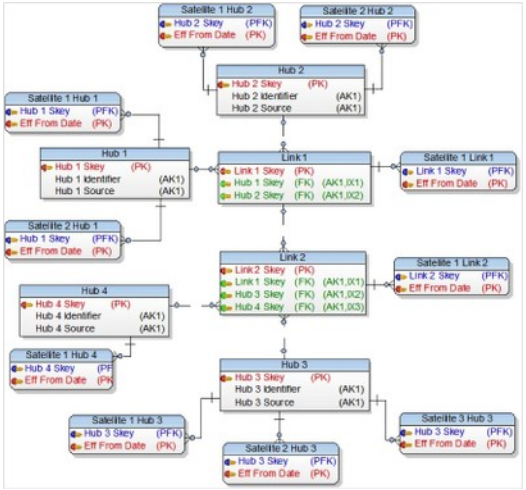
Modèles relationnels ou NoSQL

Graphes





Modélisation ensembliste

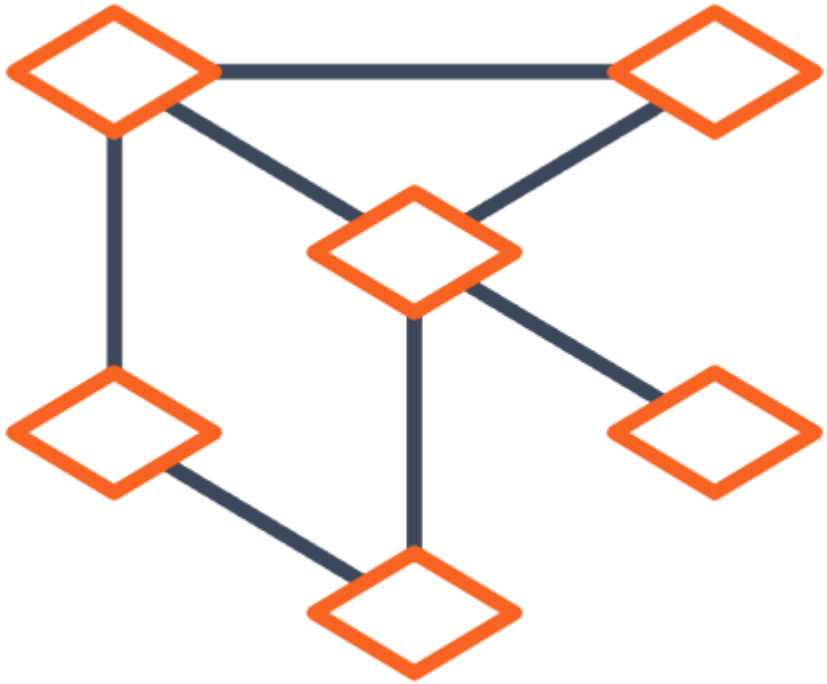


Ex. de métadonnées en graphes

Ex. de métadonnées en data vault

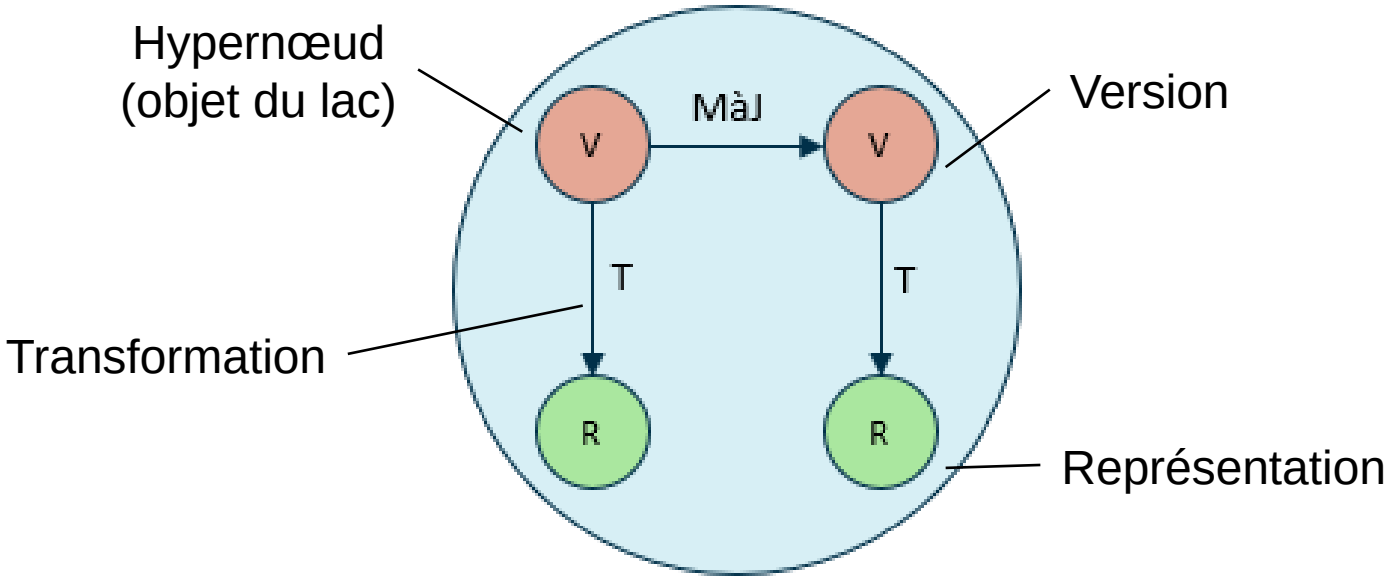
Nogueira et al. 2018

Modèles de métadonnées récents



blog.tensorflow.org

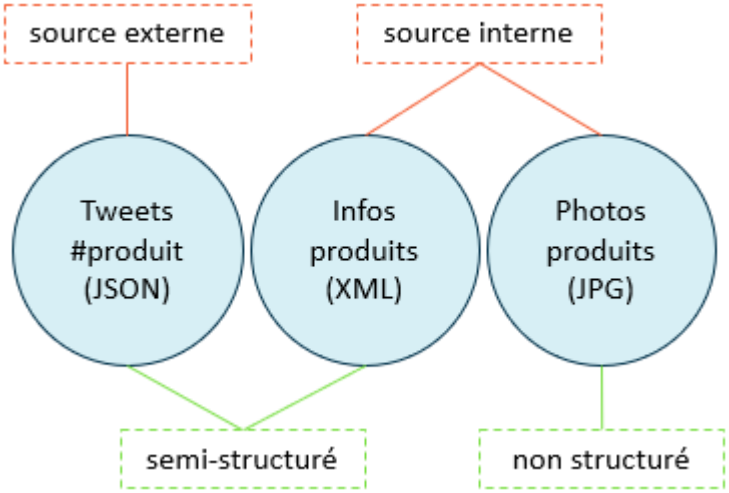
Graphe d'hypernœuds



Nœuds et arcs portent des attributs (métadonnées intra-objet).

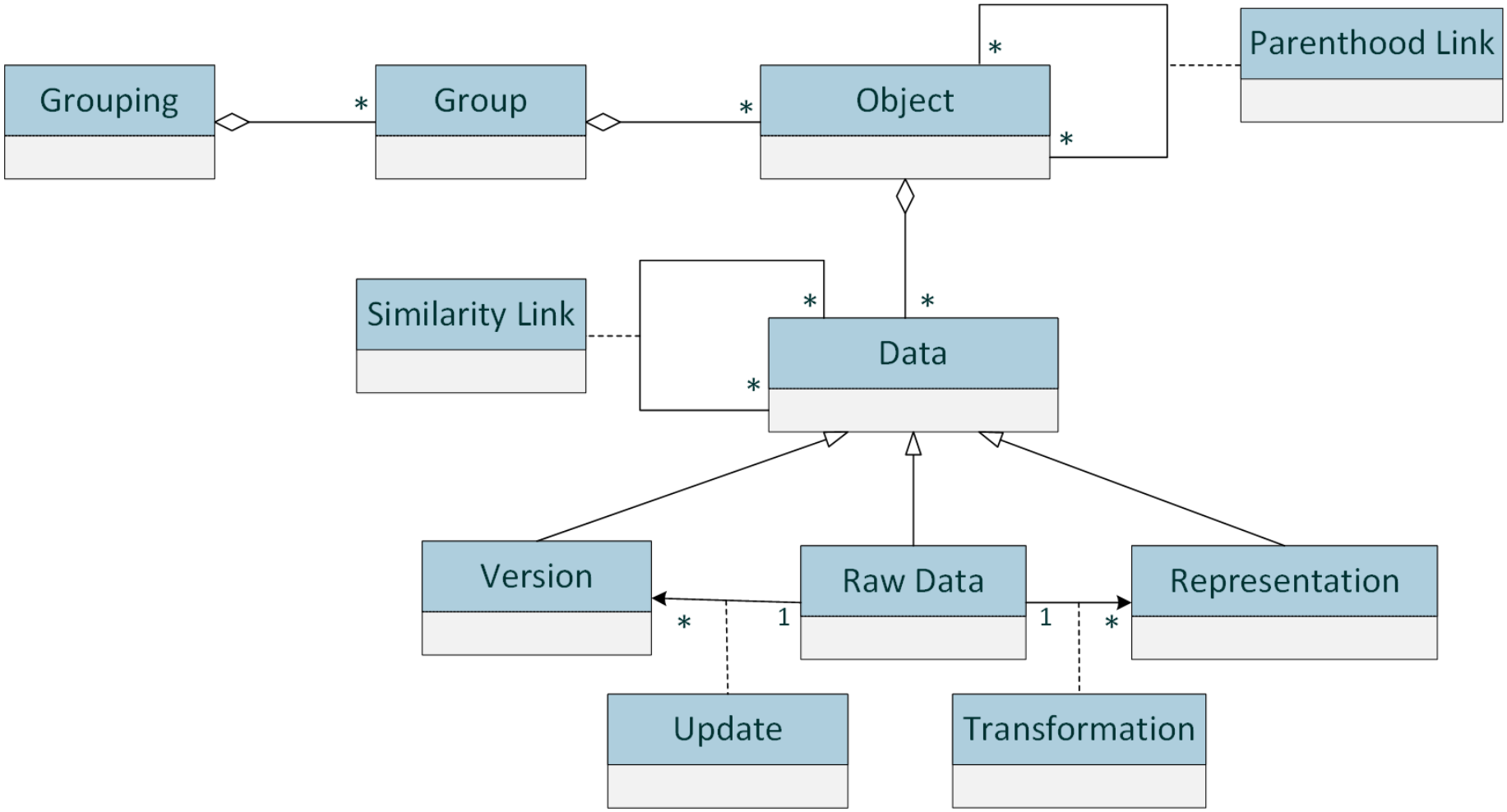
MEDAL

- Métadonnées inter-objets
  - Exemple de regroupements



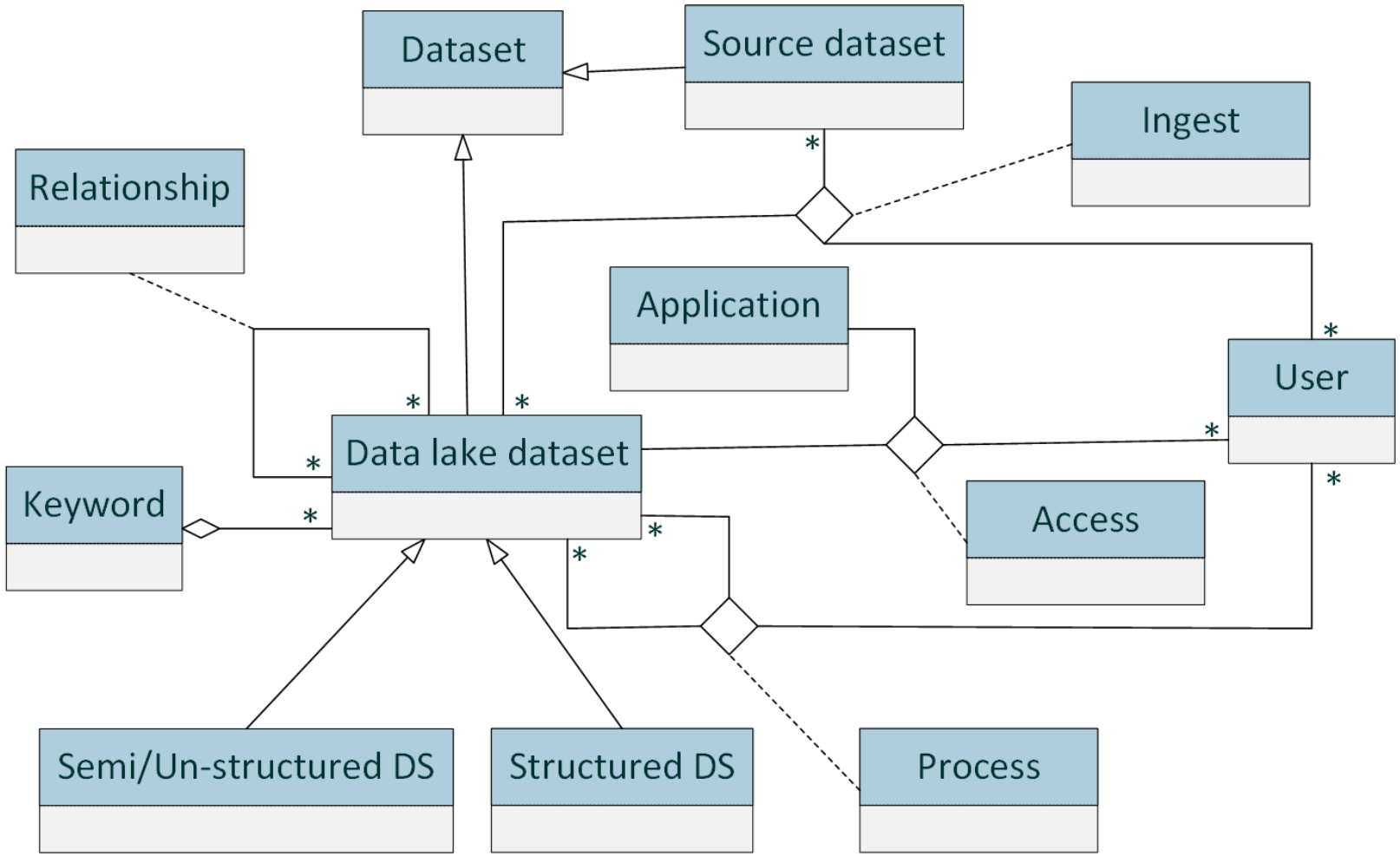
- Métadonnées globales
  - Dans des nœuds non connectés au reste du graphe

MEDAL : Modèle conceptuel

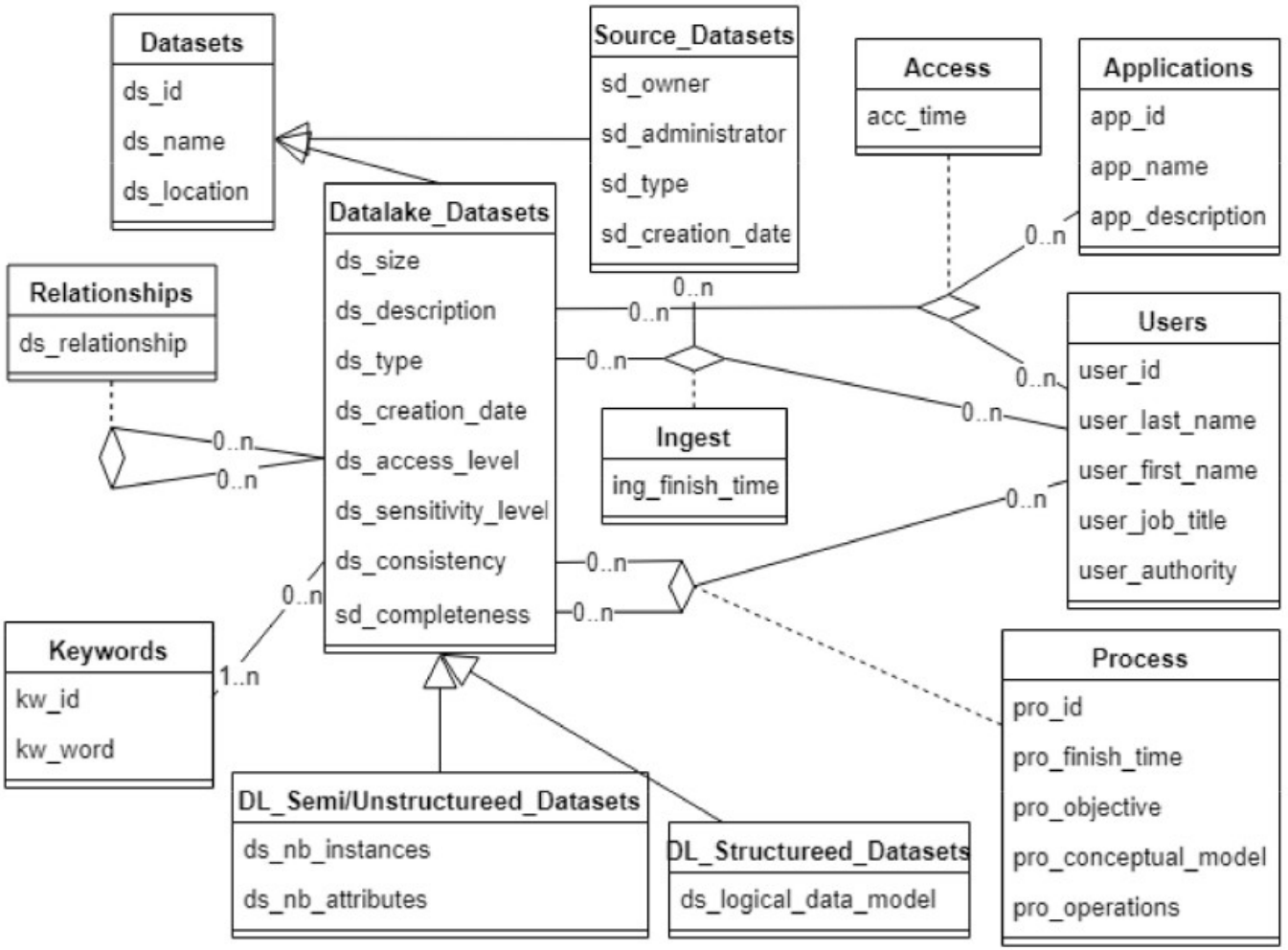


DAMMS

Ravat & Zhao 2019

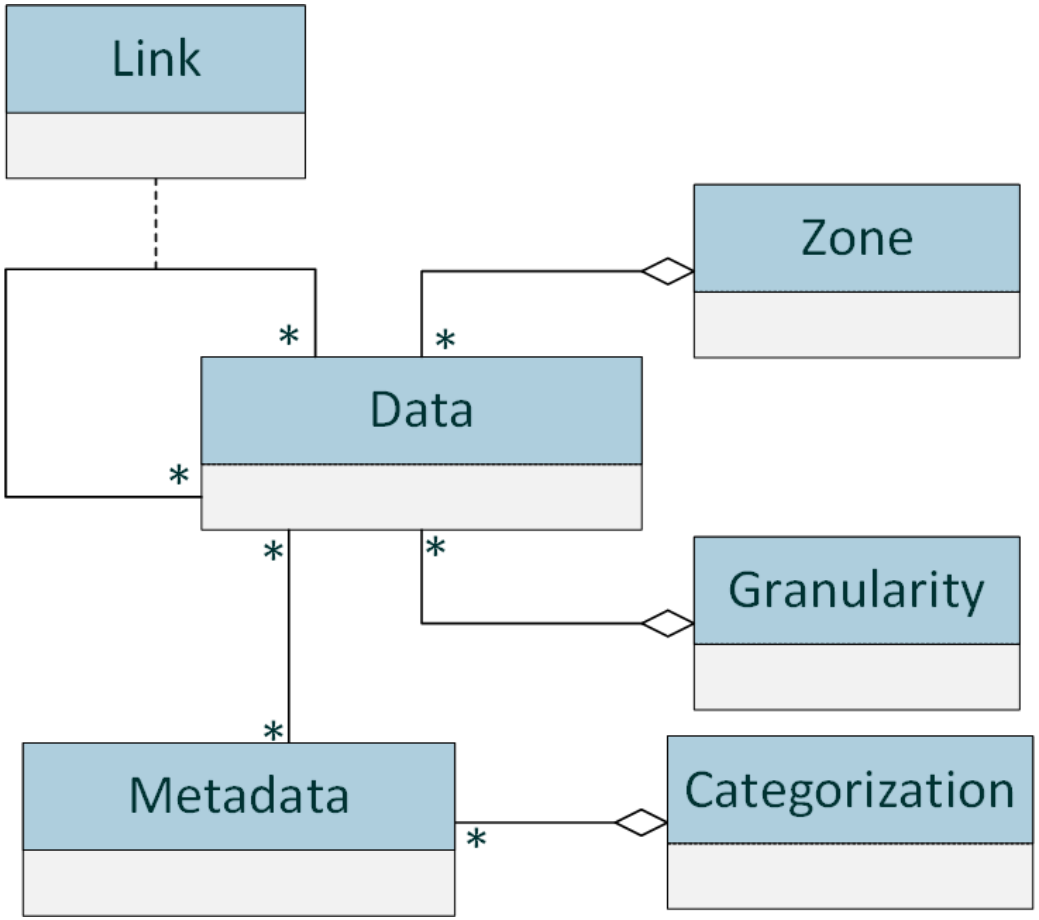


DAMMS dans le détail



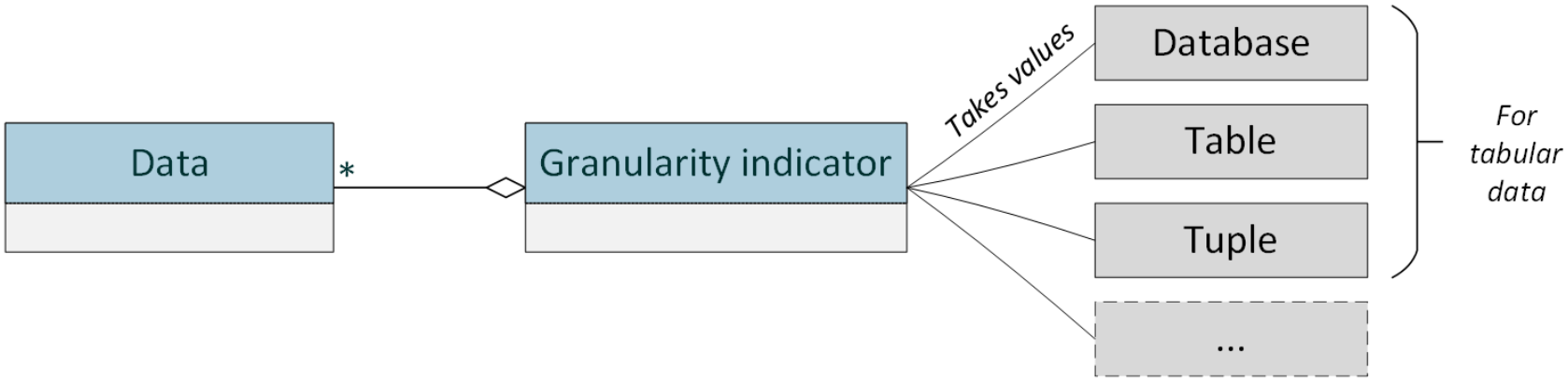
HANDLE

Eichler et al. 2020



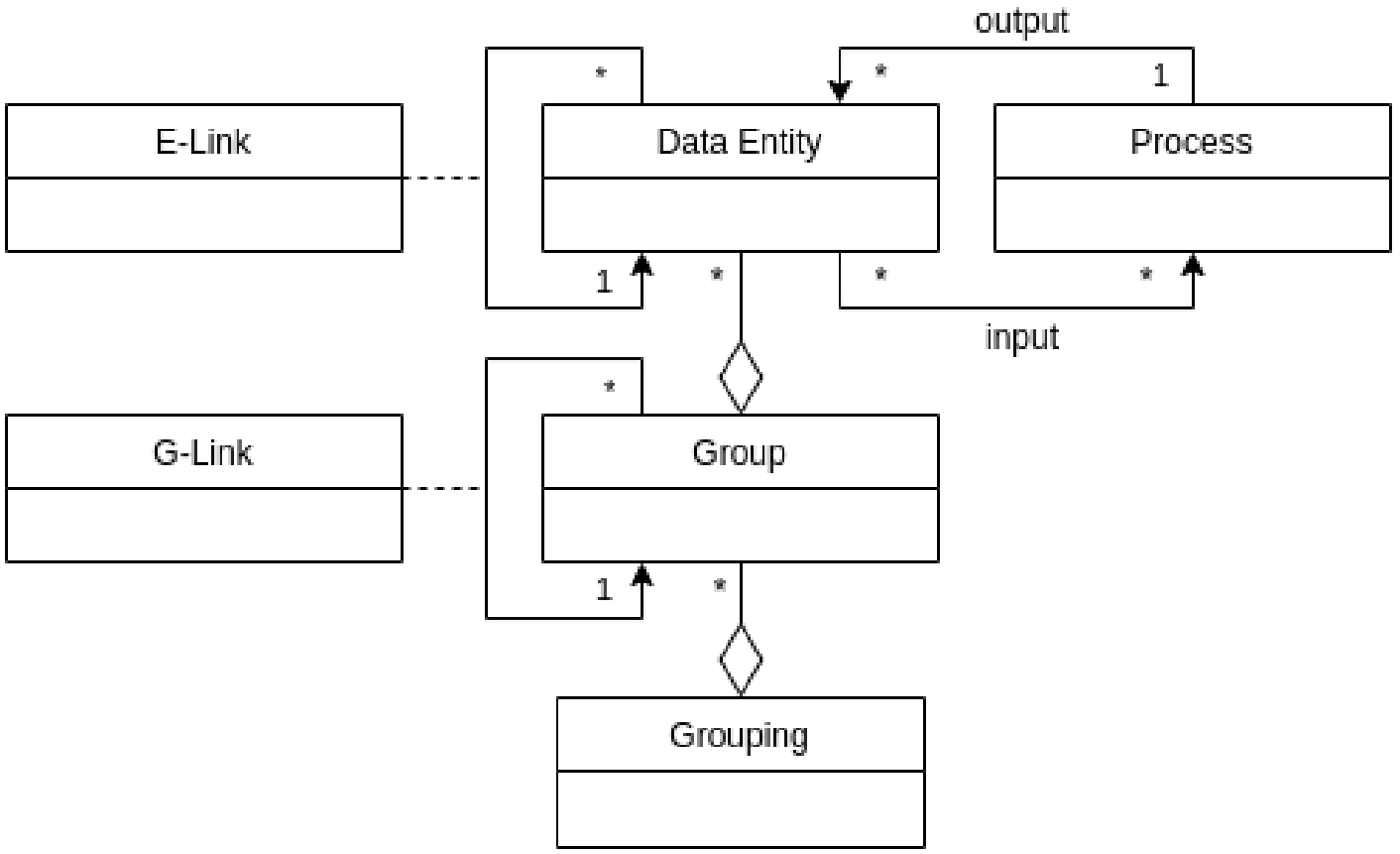
Extensions HANDLE

Exemple : extension granularités (existent aussi pour les zones et catégorisations)



goldMEDAL

Scholly et al. 2021



goldMEDAL

- Généralise les concepts des autres modèles
  - Zones et granularité à l'aide des groupements
- Possibilité explicite de modéliser le lignage
  - Avec les processus
  - Dynamique des données et des métadonnées
- Modèles conceptuels, logique et physique
- Le plus générique aujourd'hui ! (métamodèle)

Comparaison des modèles de métadonnées

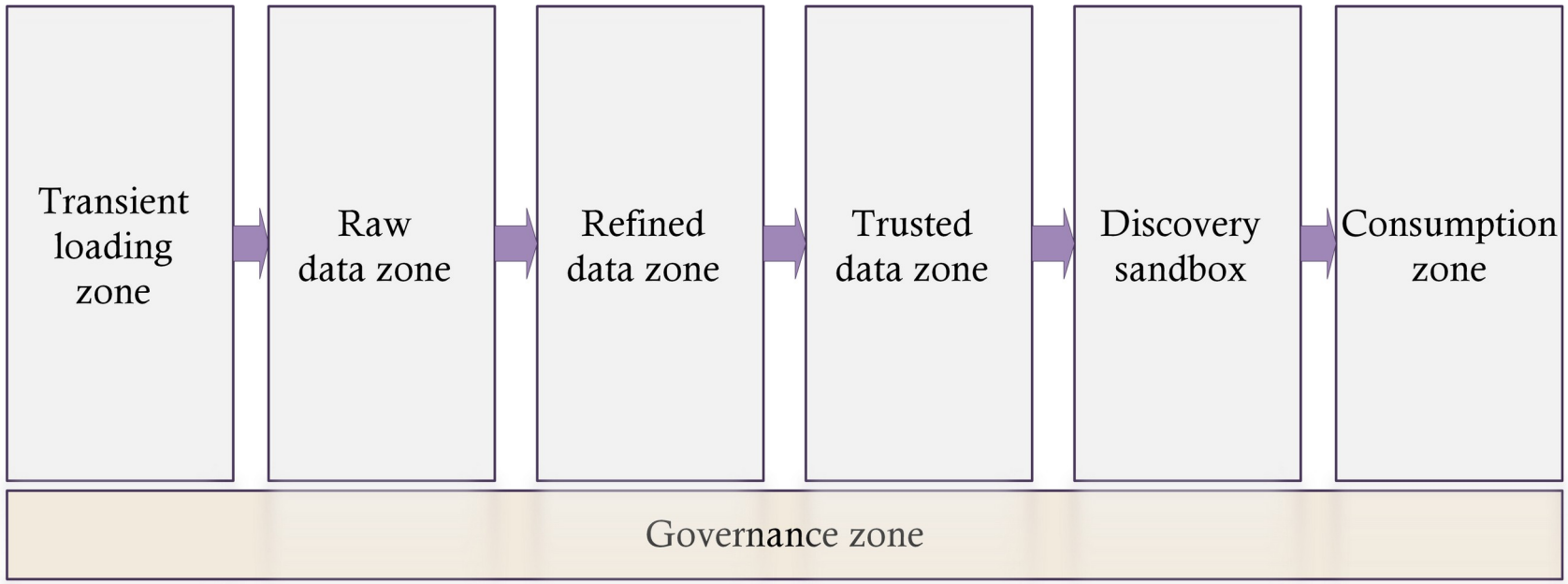
Caractéristiques	MEDAL	DAMMS	HANDLE	goldMEDAL
Enrichissement sémantique	X	X	X	X
Polymorphism/zones multiples	X	X	X	X
Versionnement des données	X	X		X
Suivi des usages	X	X	X	X
Catégorisation	X	X	X	X
Liens de similarité	X	X	X	X
Propriétés des métadonnées	X	X	X	X
Granularités multiples			X	X
Total	7/8	7/8	7/8	8/8

Plan (partie 1)

- ✓ Définitions
- ✓ Entrepôts et lacs de données
- ✓ Métadonnées et modèles de métadonnées
- Architectures et technologies pour les lacs
- Discussion, travaux de recherche

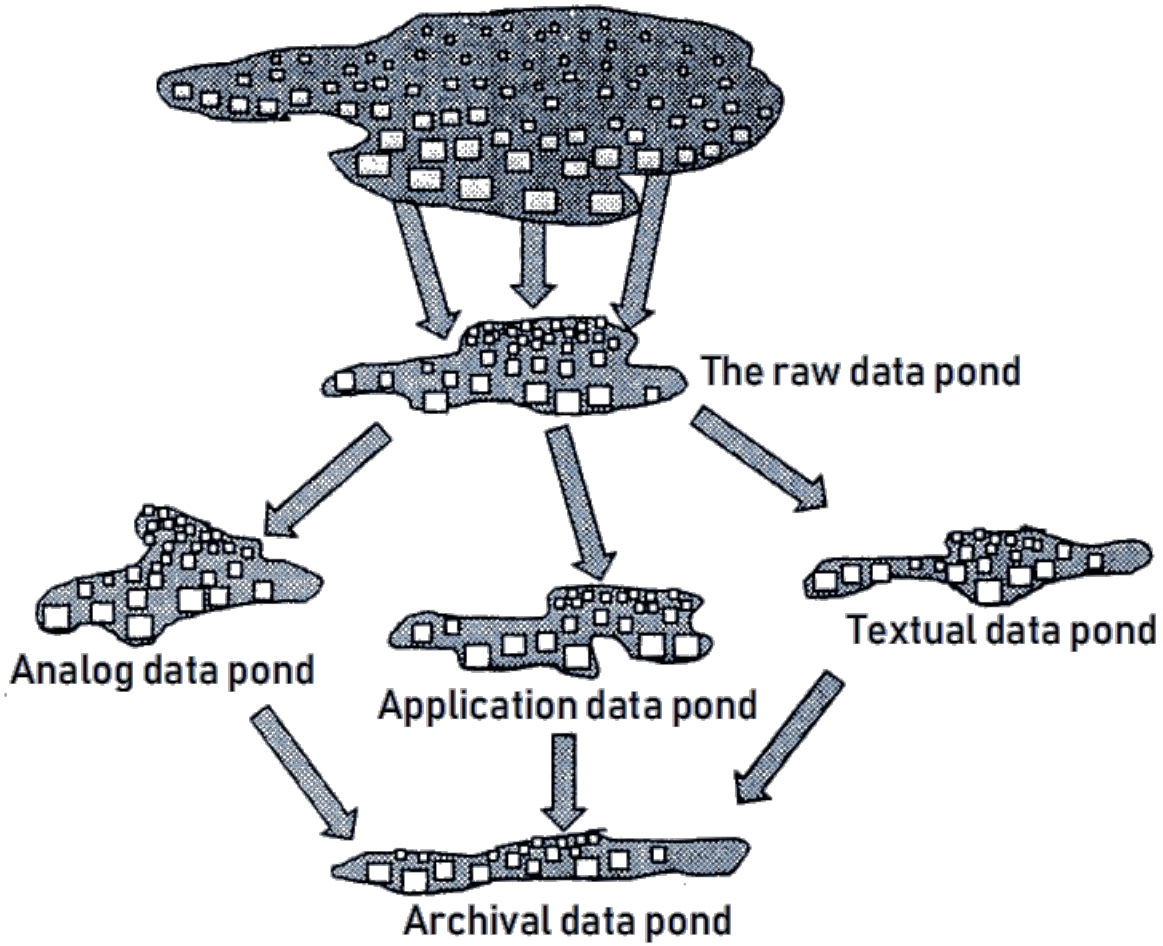
Architecture en zones

Zaloni 2015



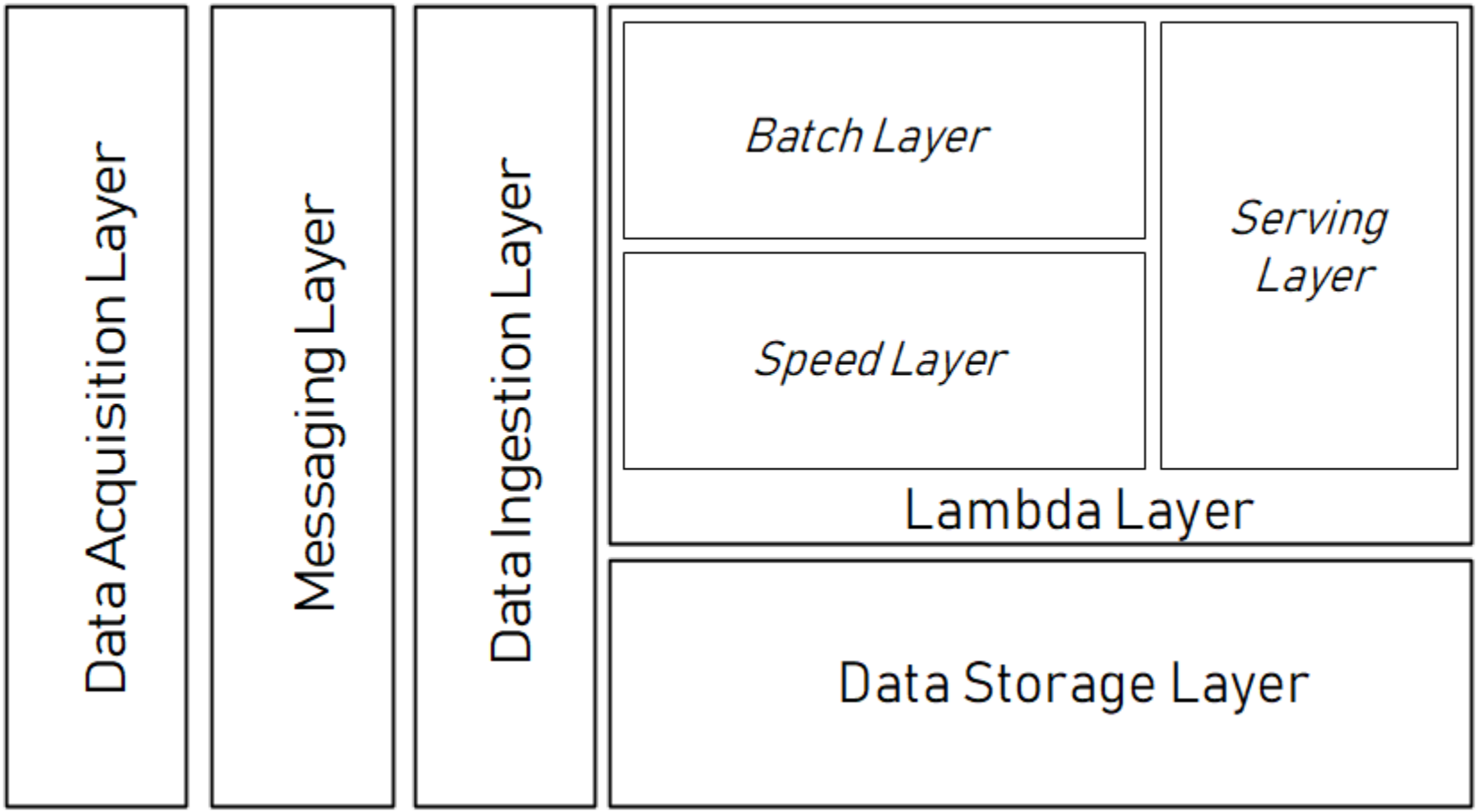
Architecture en bassins

Inmon 2016



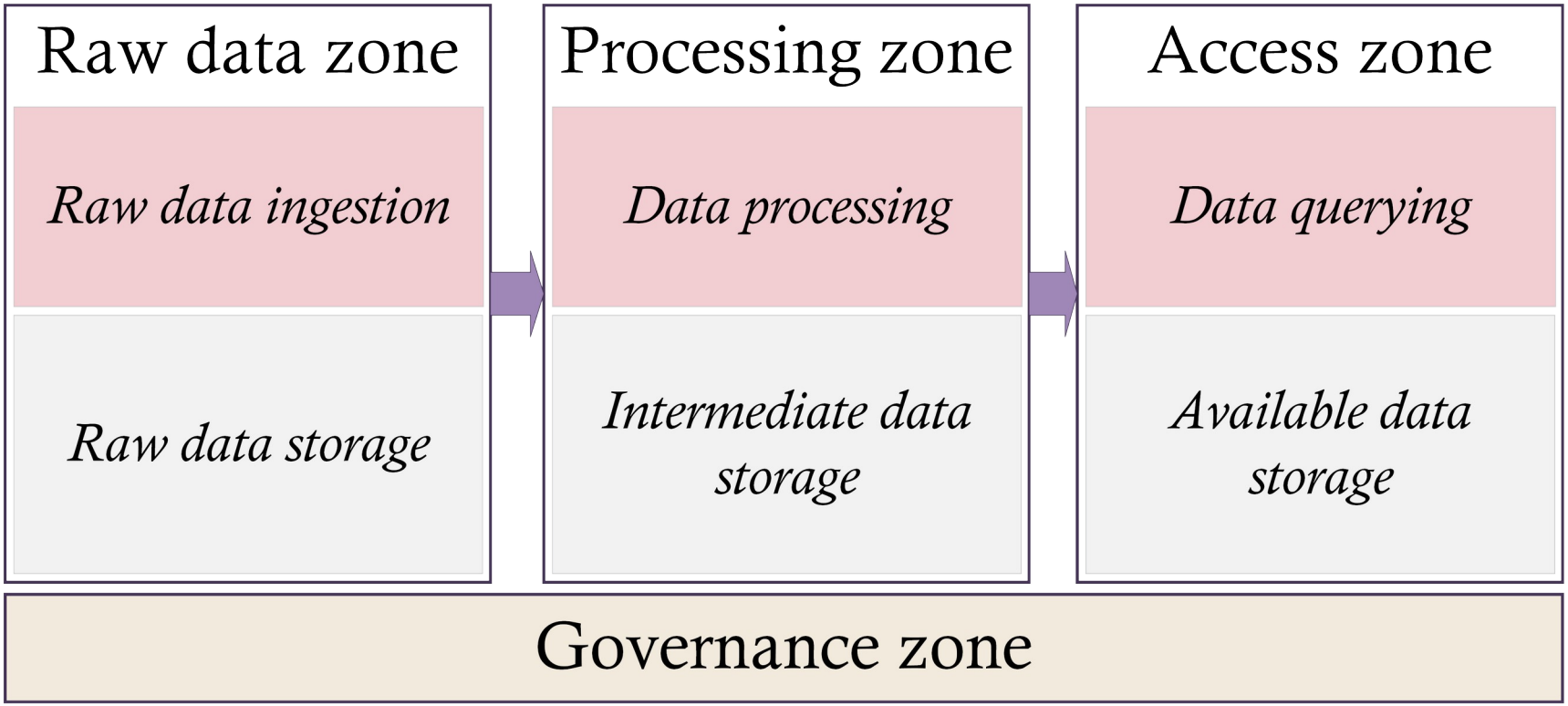
Architecture lambda

John & Misra 2017

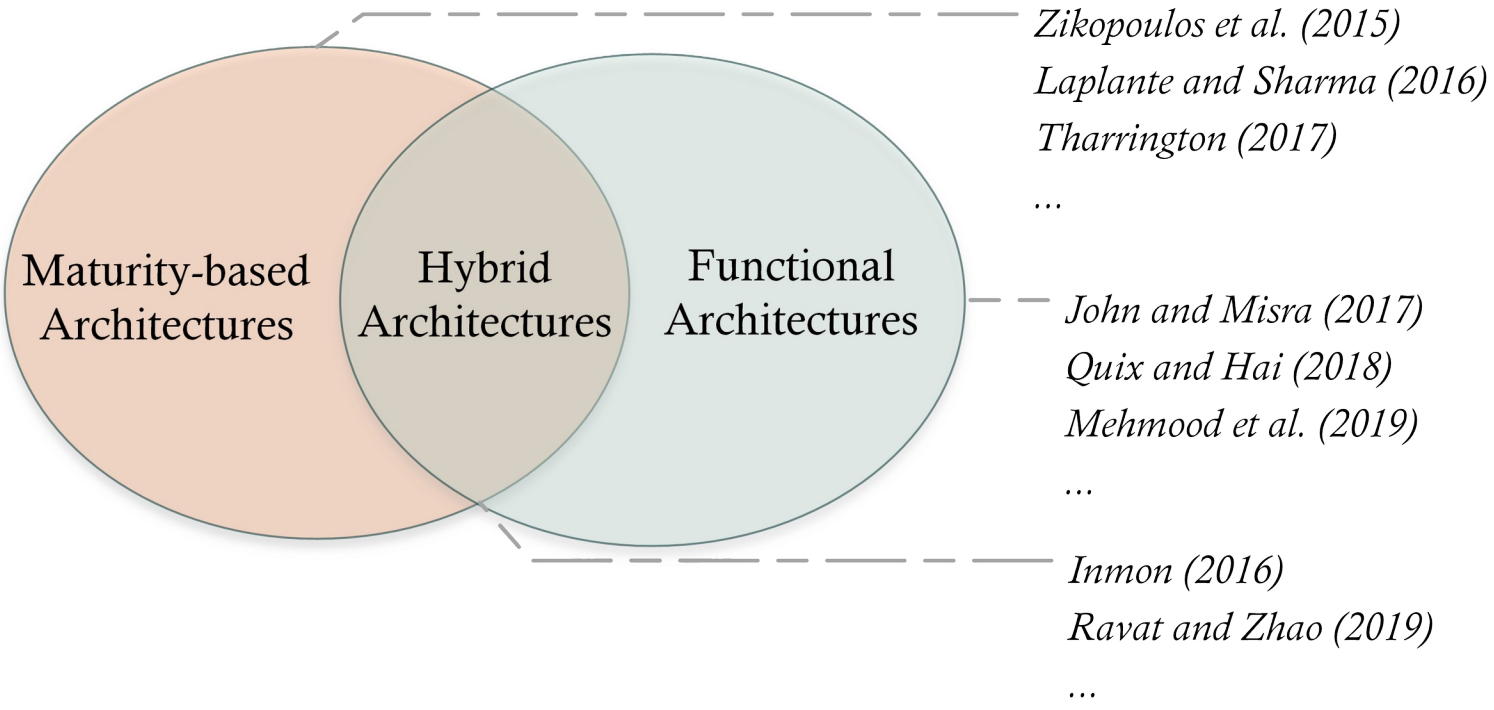


Architecture hybride

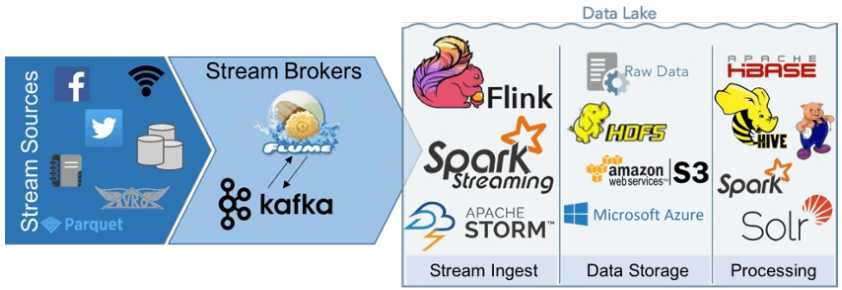
Ravat & Zhao 2019



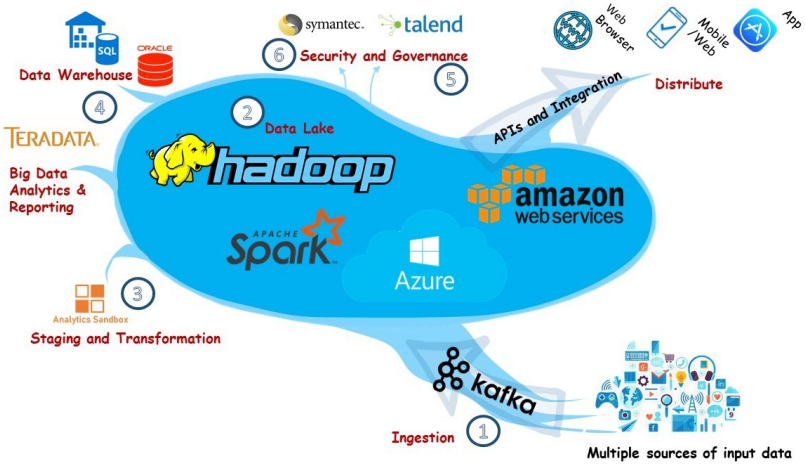
Architectures fonctionnalité × maturité



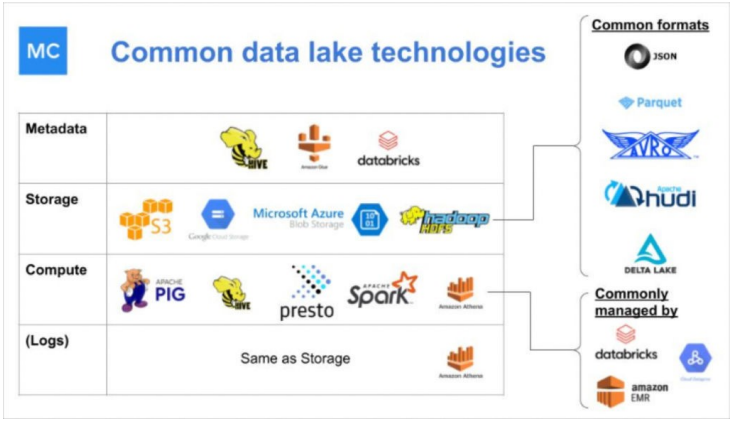
Technos pour les lacs de données



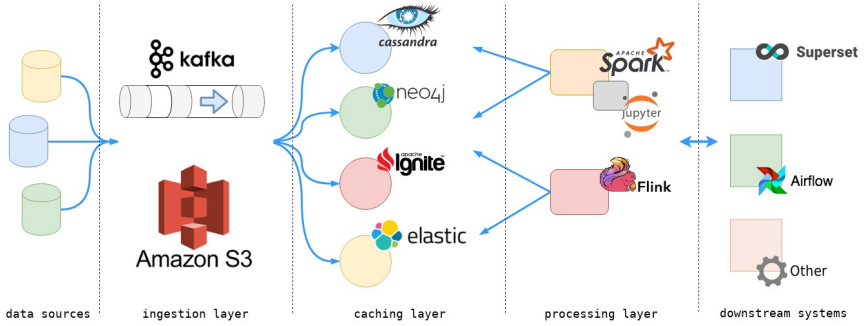
7wdata.be



kms-world.com

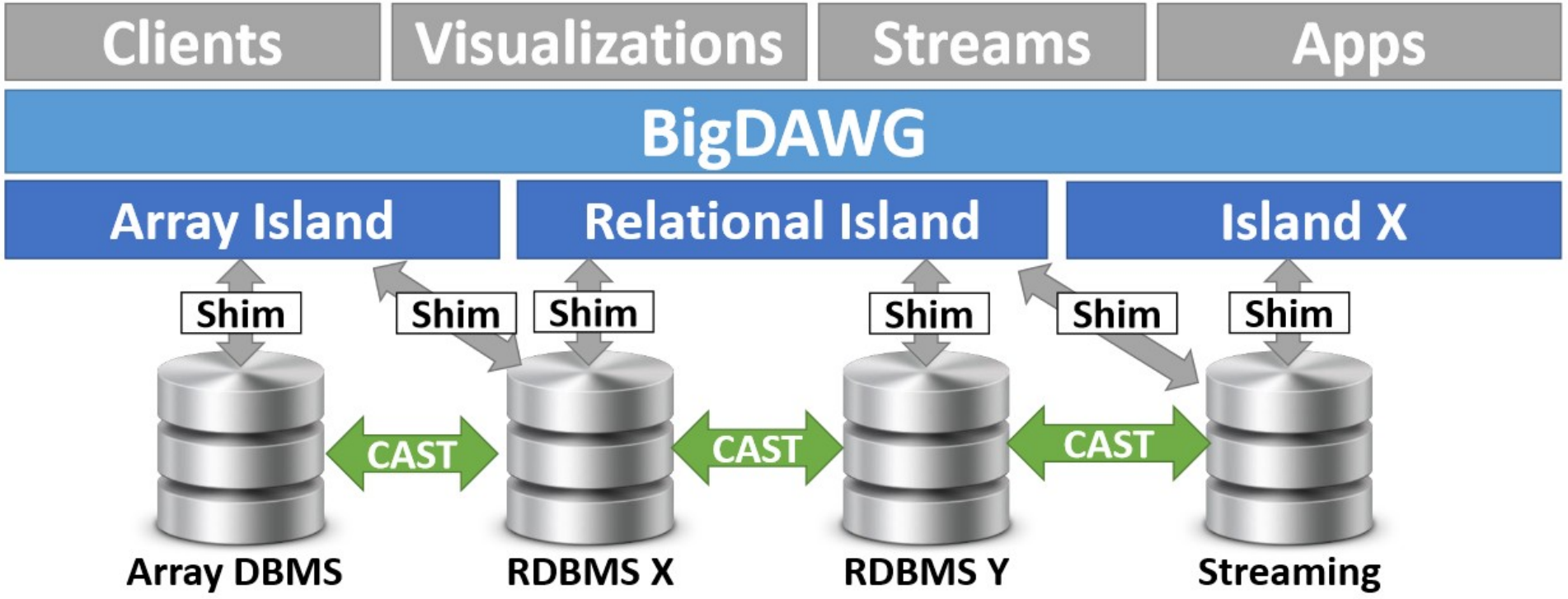


www.montecarlo.com



smartcat.io

Polystores (ex. BigDAWG)



Jennie Duggan, Aaron J. Elmore, Michael Stonebraker, Magda Balazinska, Bill Howe, Jeremy Kepner, Sam Madden, David Maier, Tim Mattson, and Stan Zdonik. 2015. The BigDAWG Polystore System. SIGMOD Rec. 44, 2 (August 2015), 11-16. DOI=http://dx.doi.org/10.1145/2814710.2814713

Ingestion et stockage des métadonnées

Ingestion des données

Métadonnées de base

Apache Atlas

Extraction avancée et gestion des métadonnées

Gestion des métadonnées

Tout SGBD relationnel ou NoSQL

Les spécialistes

Plan (partie 1)

- ✓ Définitions
- ✓ Entrepôts et lacs de données
- ✓ Métadonnées et modèles de métadonnées
- ✓ Architectures et technologies pour les lacs
- Discussion, travaux de recherche

Les lacs de données sur le grill

	
Agilité et flexibilité	Incompatibilité avec des méthodes
Faible coût de stockage	Incohérences de données
Fidélité aux données	Confusion autour du concept de lac
Gestion de données non structurées	Besoin d'interfaces d'accès données
Intégration des données en temps réel	Besoin de métadonnées adaptées
Détection de relations entre données	Manque de standards de gouvernance
Analyses à la volée	Compétences pour la mise en œuvre
Passage à l'échelle	Sécurité en maturation
Tolérance aux pannes	Monolithiques, « challengés » par les
Exploitation de données tierces	maillages de données (*data meshes*)

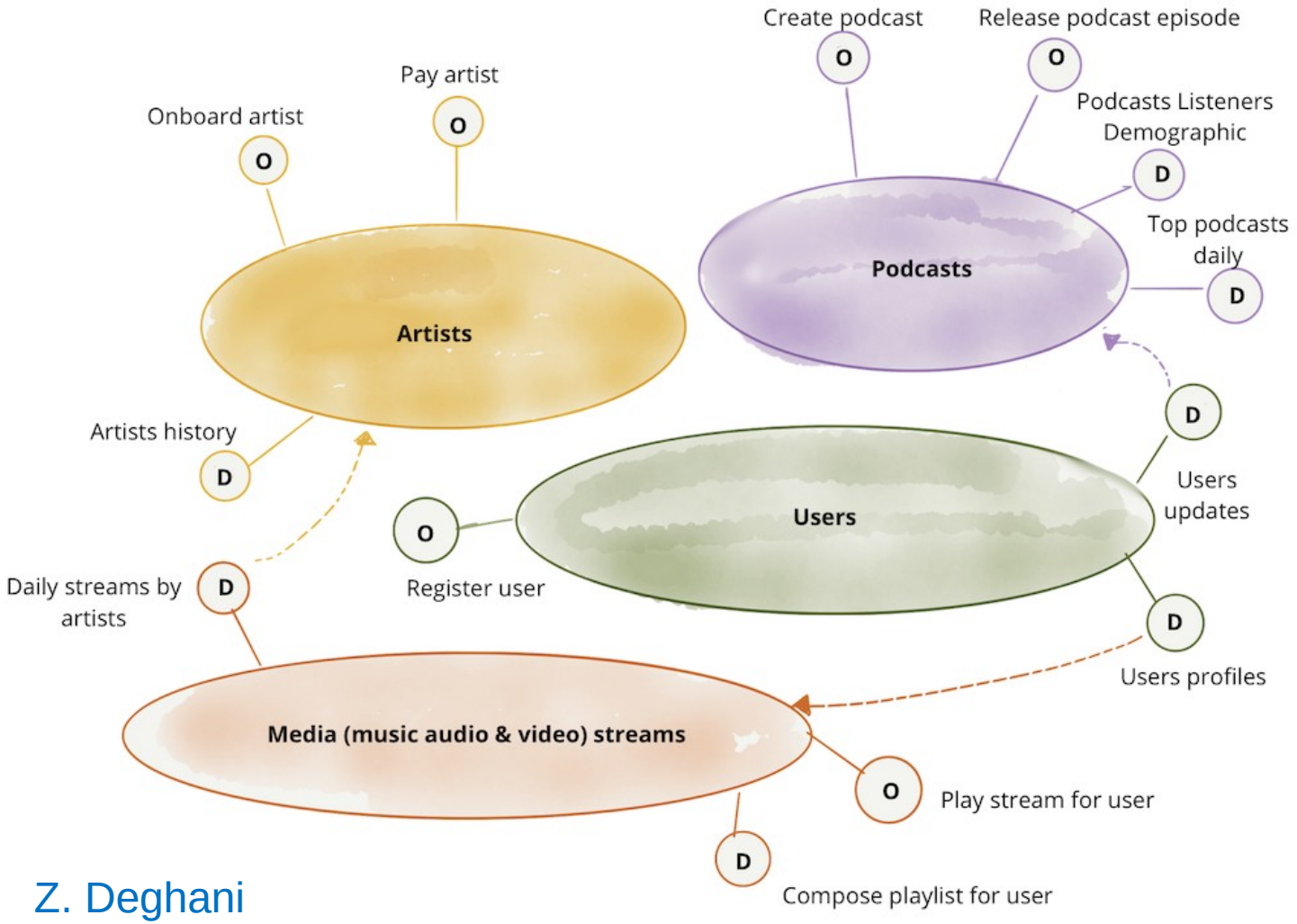
Data mesh ou la décentralisation

Zhamak Dehghani 2019 (consultante principale, Thoughtworks)

- Objectif : passage à l'échelle
  - Changement rapide des données et de leurs modèles
  - Croissance permanente des producteurs de données
  - Augmentation des consommateurs de données (ML)

- Les 4 piliers du data mesh
  - Découpage en domaines de données
  - Gestion des données en tant que produits
  - Infrastructure « self-service »
  - Gouvernance fédérée des données

Domaines : producteurs *et* consommateurs de données



Problèmes de recherches actuels

- Intégration/transformation des données
  - Optimisation des *User-Defined Functions* (UDFs)
    - Ex. Tâches MapReduce
  - Ingestion en temps réel de données à haute vélocité

- Interrogation des données
  - Interopérabilité entre données (semi-)structurées et non-structurées
  - Gravité des données, intégration virtuelle
  - Performance

Problèmes de recherches actuels

- Données non structurées
  - Génération de métadonnées
  - Données multimédia
- Gouvernance des données
  - Risques *vs.* problèmes à régler
  - Transformer les principes en solutions effectives
- Confidentialité des données
  - GDPR, quelqu'un ?

Problèmes de recherches actuels

- « Industrialisation » des lacs de données
  - Pour des utilisateurs non spécialistes
  - Couche logicielle intermédiaire *et plus !*
- Données
  - Faciles à trouver, Accessibles, Interopérables, Réutilisables (principes FAIR)
- Maintenance des métadonnées
  - Évolution de catégories, de volume, de technos...

Projets de recherche

Projet TECTONIQ
(2015)

Stage 2015
N. Pathinara

TER 2017
I. Nogueira et
M. Romdane

Projet COREL
(2017-2018)

Stages 2018
P. Sawadogo
et T. Kibata

Projet AURA-PMI
(2017-2021)

Thèse 2018-2021
P. Sawadogo

Projet STRATEGE
(2019)

Stage 2019
R. Dib

Projet HyperThesau
(2018-2019)

Postdoc P. Liu
2019-2020

Projet LIFRANUM
(2020-2023)

Postdoc
J. Espinosa
2020-2021

Stage
V. Renault
2022

Projets de recherche

Projet DataLAC
(2020-2023)

Stage
L. Ciuraneta
2021

Thèse CIFRE
2022-2025
A. Diouane

Projet
PicassoLetters
(2021-2022)

Stage
I. Slalmi
2022

Stage
R. El-Idrissi
2023

Thèse
R. El-Idrissi
2023-2026

Stage
A. Derder
2023

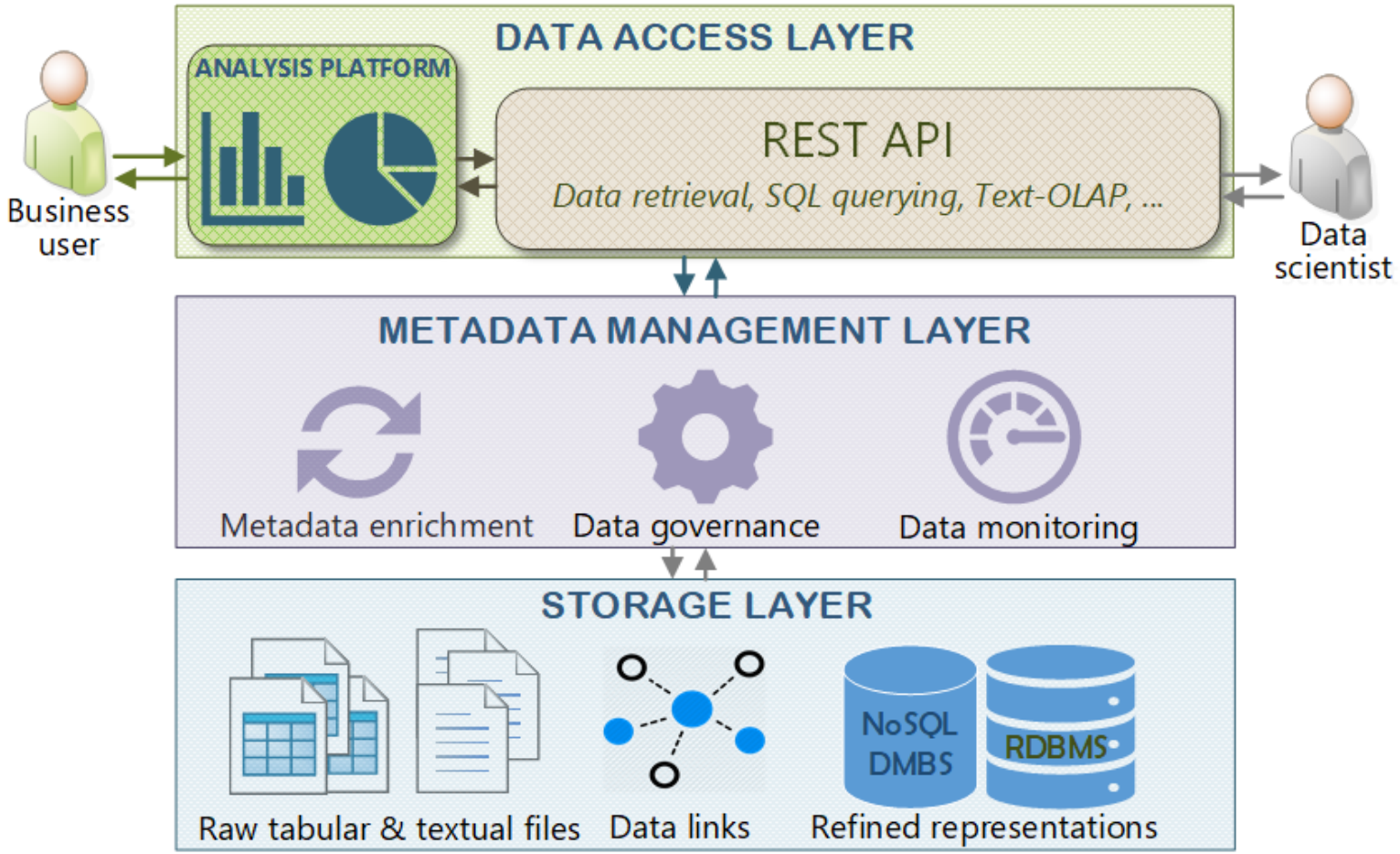
Stage
R. Aoudj
2023

Réseau de recherche *datalakers*

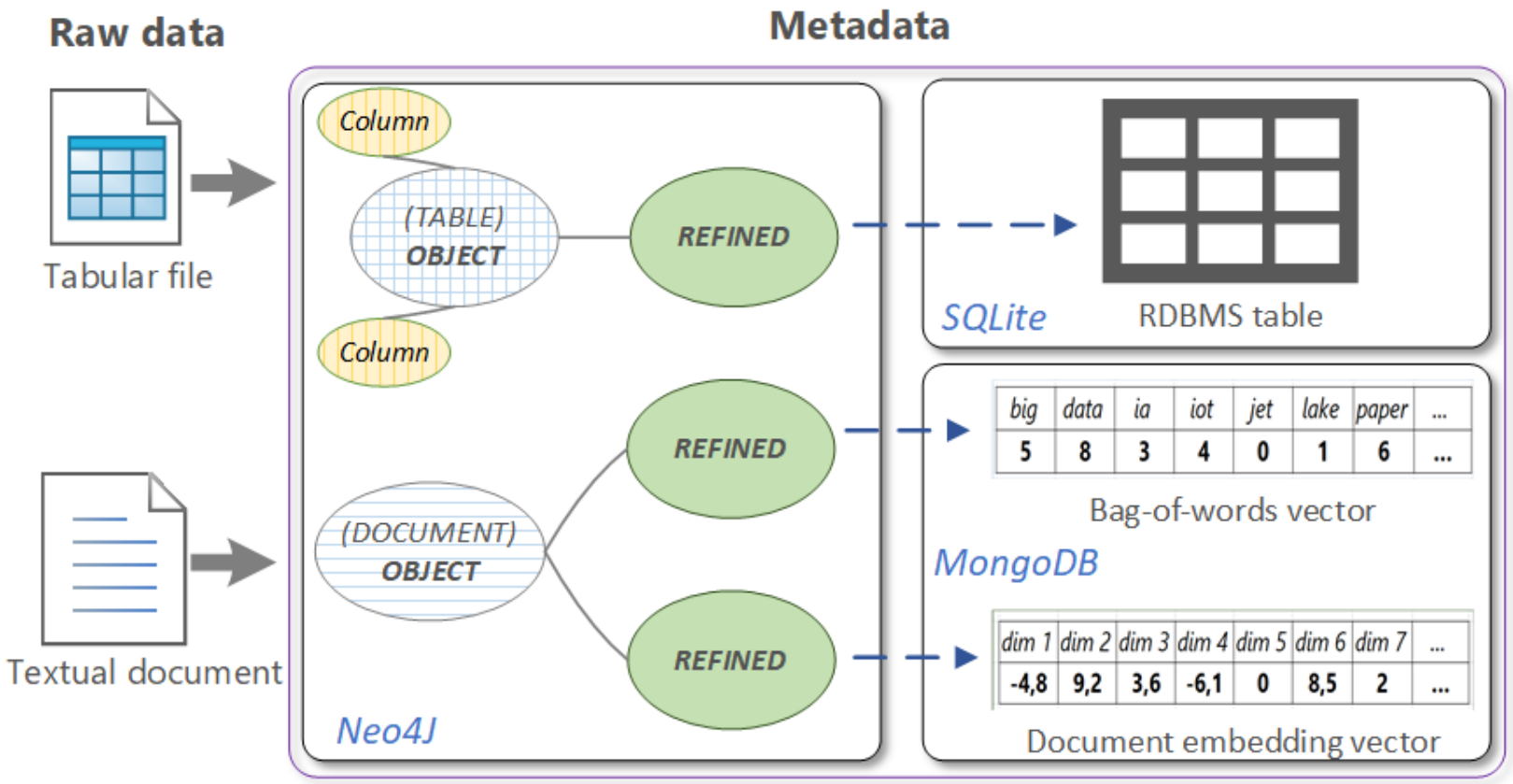
S. Loudcher

Exemple : AUDAL

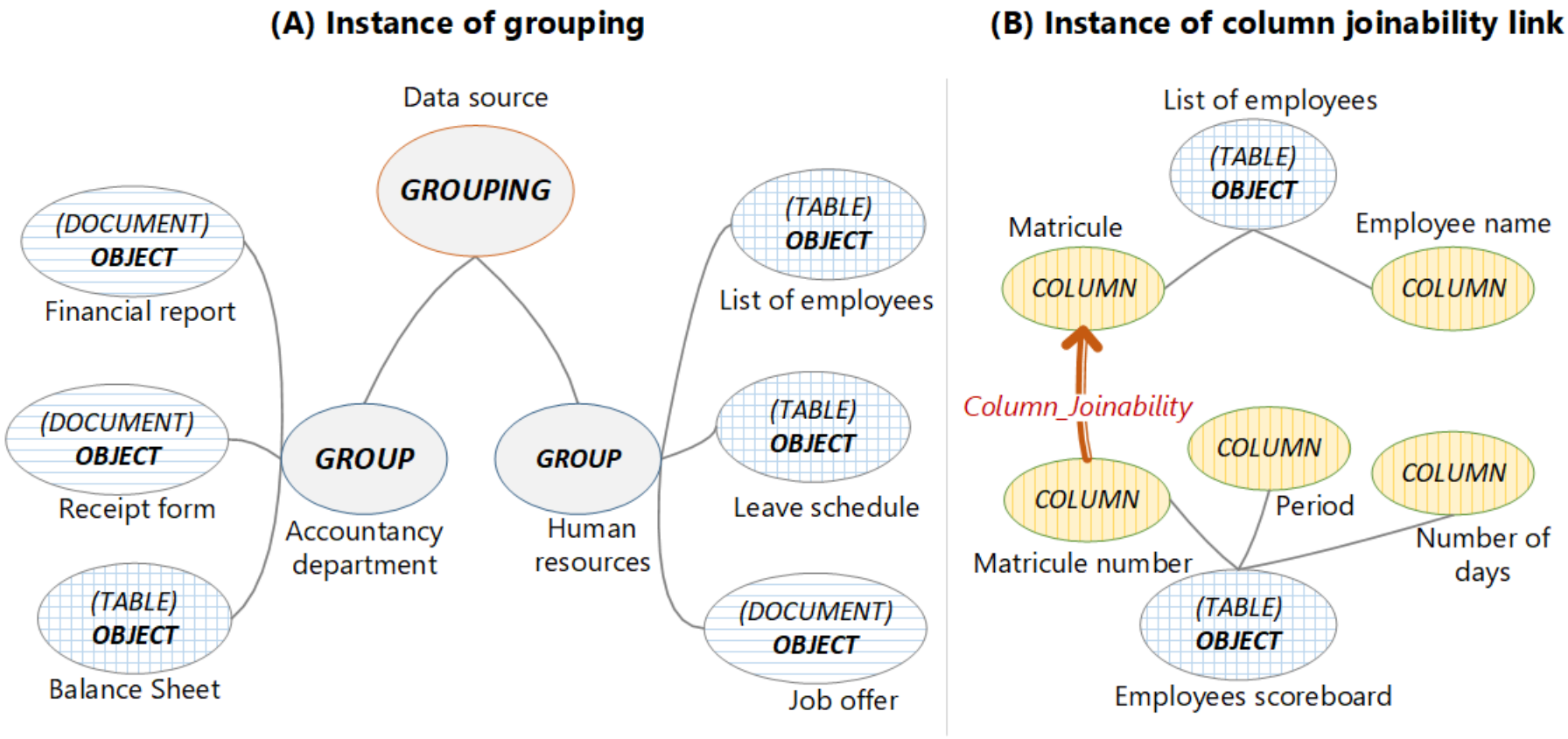
Sawadogo et Darmont 2021

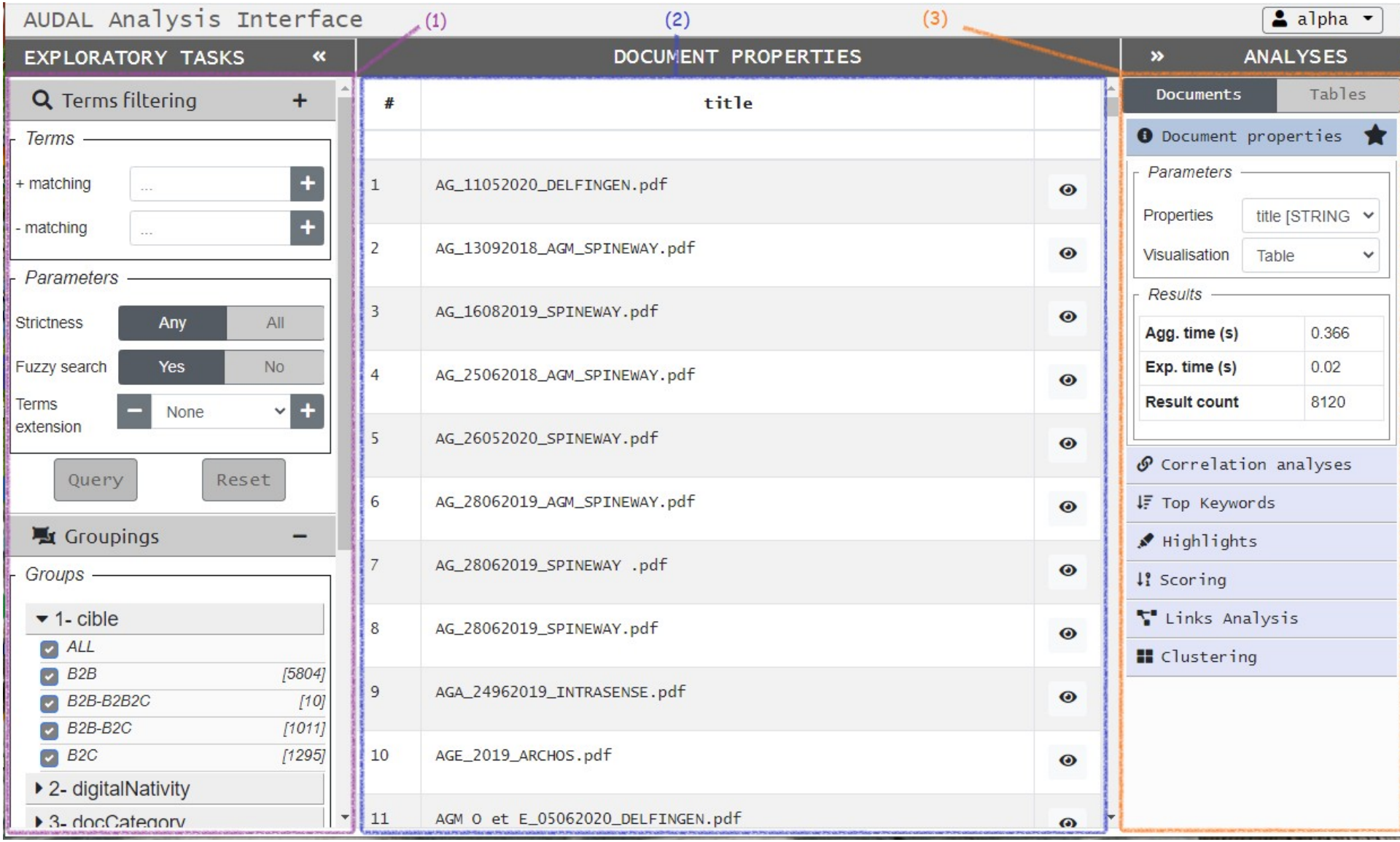


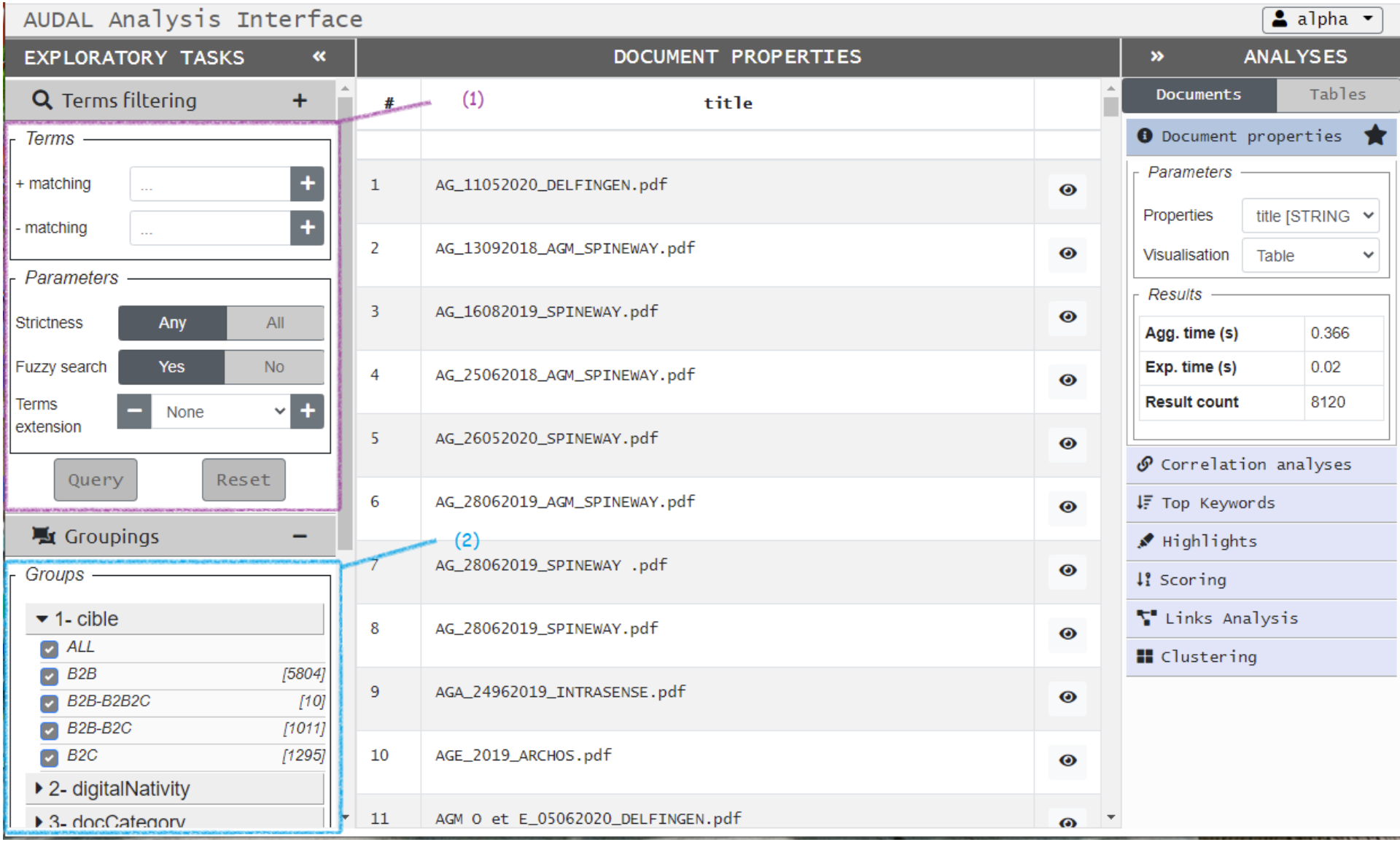
Métadonnées intra-objet d'AUDAL

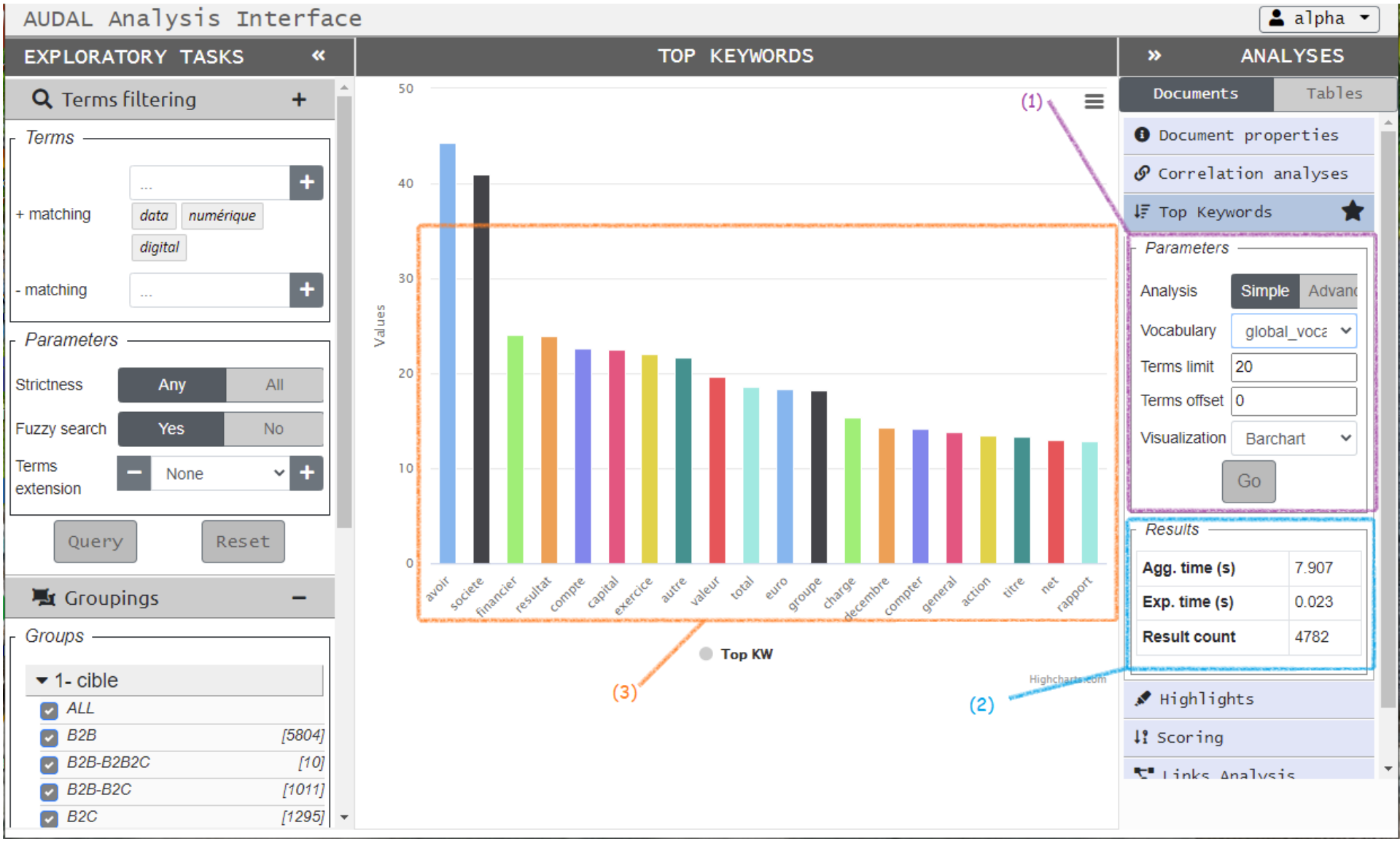


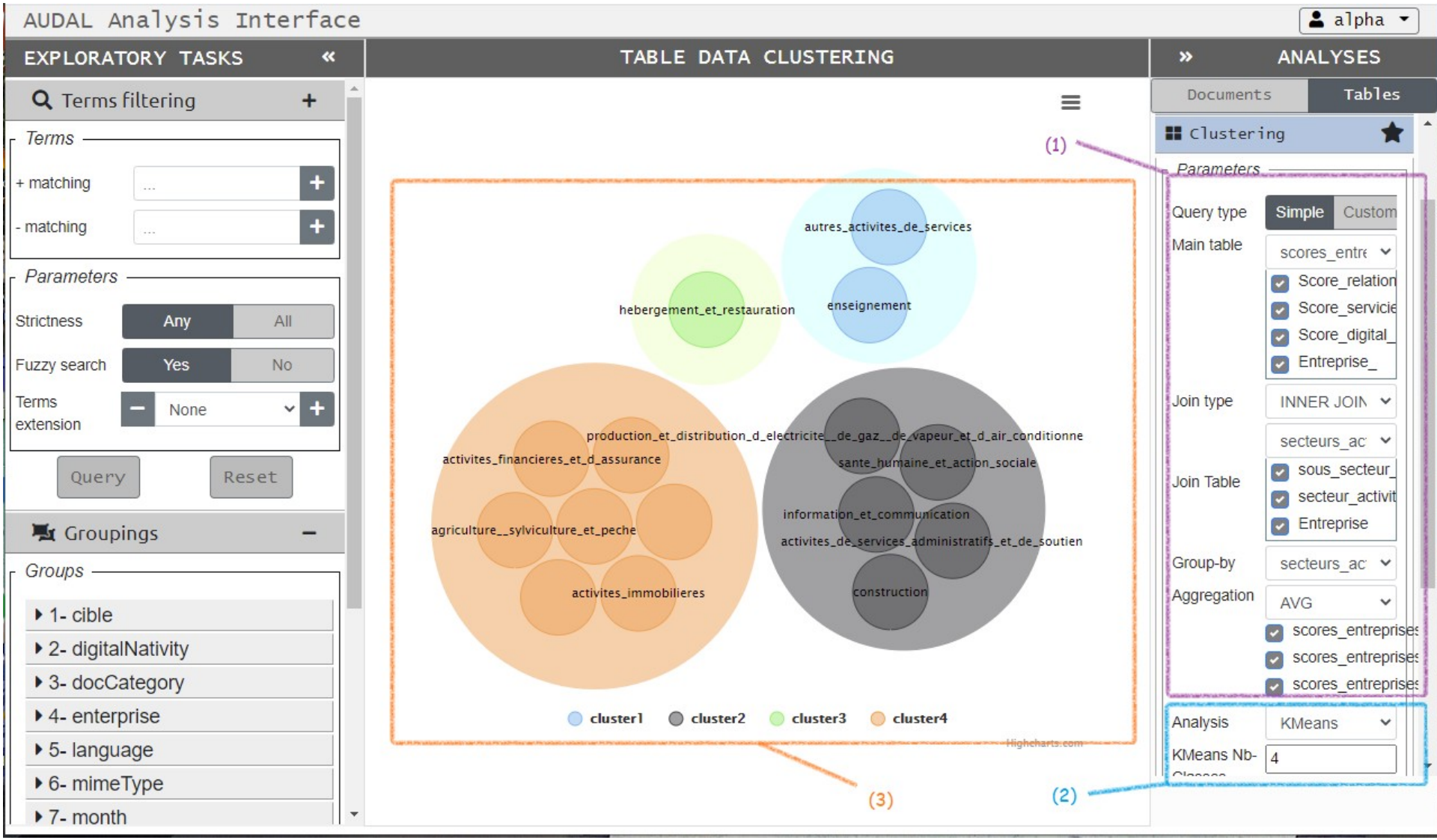
Métadonnées inter-objets d'AUDAL




AUDAL Analysis Interface
(1)
(2)
(3)
alpha
EXPLORATORY TASKS
DOCUMENT PROPERTIES
ANALYSES
Terms filtering
Terms
+ matching
- matching
Parameters
Strictness
Any
All
Fuzzy search
Yes
No
Terms extension
None
Query
Reset
Groupings
Groups
1- cible
ALL
B2B [5804]
B2B-B2B2C [10]
B2B-B2C [1011]
B2C [1295]
2- digitalNativity
3- docCategory
title
1 AG_11052020_DELFINGEN.pdf
2 AG_13092018_AGM_SPINEWAY.pdf
3 AG_16082019_SPINEWAY.pdf
4 AG_25062018_AGM_SPINEWAY.pdf
5 AG_26052020_SPINEWAY.pdf
6 AG_28062019_AGM_SPINEWAY.pdf
7 AG_28062019_SPINEWAY .pdf
8 AG_28062019_SPINEWAY.pdf
9 AGA_24962019_INTRASENSE.pdf
10 AGE_2019_ARCHOS.pdf
11 AGM O et E_05062020_DELFINGEN.pdf
Documents
Tables
Document properties
Parameters
Properties
title [STRING
Visualisation
Table
Results
Agg. time (s) 0.366
Exp. time (s) 0.02
Result count 8120
Correlation analyses
Top Keywords
Highlights
Scoring
Links Analysis
Clustering


AUDAL Analysis Interface
alpha
EXPLORATORY TASKS
DOCUMENT PROPERTIES
ANALYSES
Terms filtering
Terms
+ matching
- matching
Parameters
Strictness
Any
All
Fuzzy search
Yes
No
Terms extension
None
Query
Reset
Groupings
Groups
1- cible
ALL
B2B [5804]
B2B-B2B2C [10]
B2B-B2C [1011]
B2C [1295]
2- digitalNativity
3- docCategory
(1)
(2)
title
1 AG_11052020_DELFINGEN.pdf
2 AG_13092018_AGM_SPINEWAY.pdf
3 AG_16082019_SPINEWAY.pdf
4 AG_25062018_AGM_SPINEWAY.pdf
5 AG_26052020_SPINEWAY.pdf
6 AG_28062019_AGM_SPINEWAY.pdf
7 AG_28062019_SPINEWAY .pdf
8 AG_28062019_SPINEWAY.pdf
9 AGA_24962019_INTRASENSE.pdf
10 AGE_2019_ARCHOS.pdf
11 AGM O et E_05062020_DELFINGEN.pdf
Documents
Tables
Document properties
Parameters
Properties
title [STRING
Visualisation
Table
Results
Agg. time (s) 0.366
Exp. time (s) 0.02
Result count 8120
Correlation analyses
Top Keywords
Highlights
Scoring
Links Analysis
Clustering


AUDAL Analysis Interface
alpha
EXPLORATORY TASKS
TOP KEYWORDS
ANALYSES
Terms filtering
Terms
+ matching
data
numérique
digital
- matching
Parameters
Strictness
Any
All
Fuzzy search
Yes
No
Terms extension
None
Query
Reset
Groupings
Groups
1- cible
ALL
B2B [5804]
B2B-B2B2C [10]
B2B-B2C [1011]
B2C [1295]
Values
50
40
30
20
10
0
avoir
societe
financier
resultat
compte
capital
exercice
autre
valeur
total
euro
groupe
charge
decembre
compter
general
action
titre
net
rapport
Top KW
Highcharts.com
(1)
(2)
(3)
Documents
Tables
Document properties
Correlation analyses
Top Keywords
Parameters
Analysis
Simple
Advanc
Vocabulary
global_voca
Terms limit
20
Terms offset
0
Visualization
Barchart
Go
Results
Agg. time (s)
7.907
Exp. time (s)
0.023
Result count
4782
Highlights
Scoring
Links Analysis


AUDAL Analysis Interface
alpha
EXPLORATORY TASKS
TABLE DATA CLUSTERING
ANALYSES
Terms filtering
Terms
+ matching
- matching
Parameters
Strictness
Any
All
Fuzzy search
Yes
No
Terms extension
None
Query
Reset
Groupings
Groups
1- cible
2- digitalNativity
3- docCategory
4- enterprise
5- language
6- mimeType
7- month
autres_activites_de_services
hebergement_et_restauration
enseignement
production_et_distribution_d_electricite__de_gaz__de_vapeur_et_d_air_conditionne
activites_financieres_et_d_assurance
sante_humaine_et_action_sociale
information_et_communication
agriculture__sylviculture_et_peche
activites_de_services_administratifs_et_de_soutien
activites_immobilieres
construction
cluster1
cluster2
cluster3
cluster4
Highcharts.com
(1)
(2)
(3)
Documents
Tables
Clustering
Parameters
Query type
Simple
Custom
Main table
scores_entre
Score_relation
Score_servicie
Score_digital_
Entreprise_
Join type
INNER JOIN
secteurs_ac
Join Table
sous_secteur_
secteur_activit
Entreprise
Group-by
secteurs_ac
Aggregation
AVG
scores_entreprises
scores_entreprises
scores_entreprises
Analysis
KMeans
KMeans Nb-
4

Plan (partie 1)

- ✓ Définitions
- ✓ Entrepôts et lacs d
- ✓ Métadonnées et étadonnées
- ✓ Architectures et te pour les lacs
- ✓ Discussion, travaux de recherche

Partie 2

SGBD pour les *big data*

Objectifs

Expérimenter avec des outils NoSQL pour les *big data*

Gérer des données non-relationnelles

Requêter des données non-relationnelles

Plan (partie 2)

- ✓ Objectifs
- Gestion de graphes avec Neo4J
- Interrogation de données JSON avec RumbleDB
- Autres SGBD NoSQL

Quand utiliser un SGBD graphe ?

Si au moins deux critères sont couverts

Données dynamiques
Données connectées
Schéma flexible
Temps réel

https://logisima.developpez.com/tutoriel/nosql/neo4j/introduction-neo4j/

Neo4J : généralités

SGBD orienté graphes
Logiciel *open source*
Développé par la société Neo4j, Inc.
Langage de requête Cypher
Version 1.0 (2010) – Version 5.10 (2023)

Concept 1 : Nœud

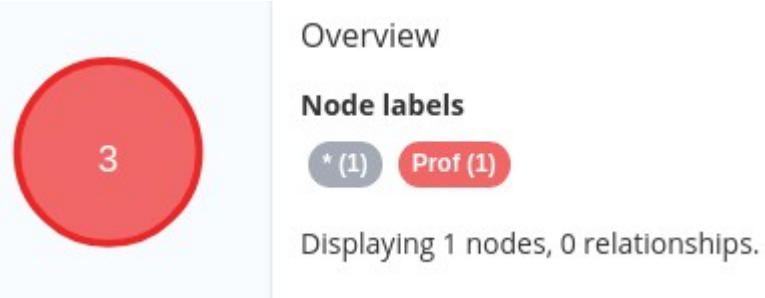
Représentation graphique

Cypher

```
// Je suis un commentaire
create (n)
return n
```

Concept 2 : Label (~type)

Représentation graphique

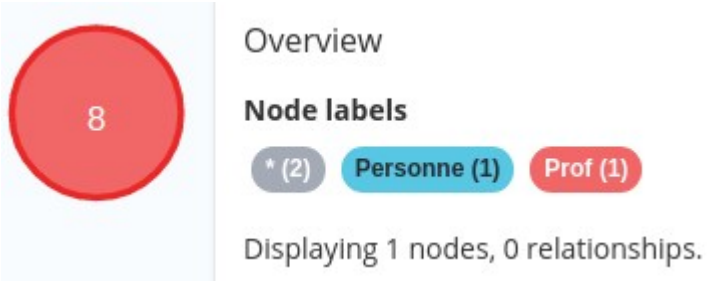


Cypher

```
create (n : Prof)
return n
```

Labels multiples

Représentation graphique

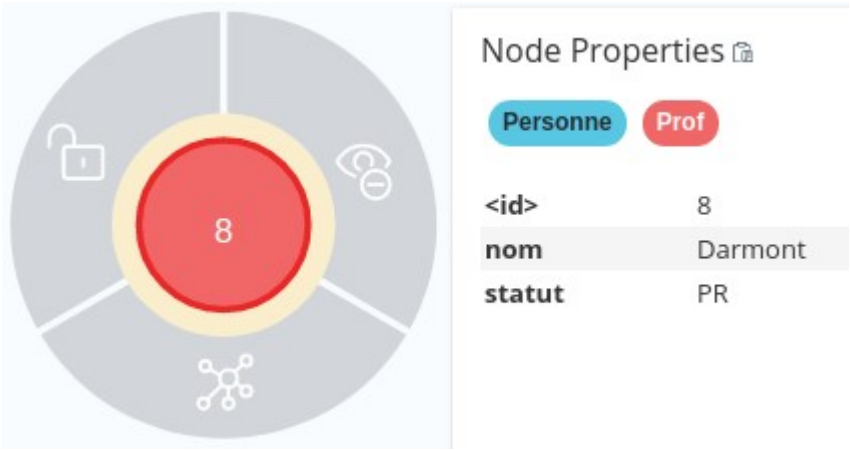


Cypher

```
create (n : Personne : Prof)
return n
```

Concept 3 : Propriétés

Représentation graphique



Cypher

```
create (n : Personne : Prof {nom : "Darmont", statut : "PR"})
return n
```

Quelques nœuds de plus !

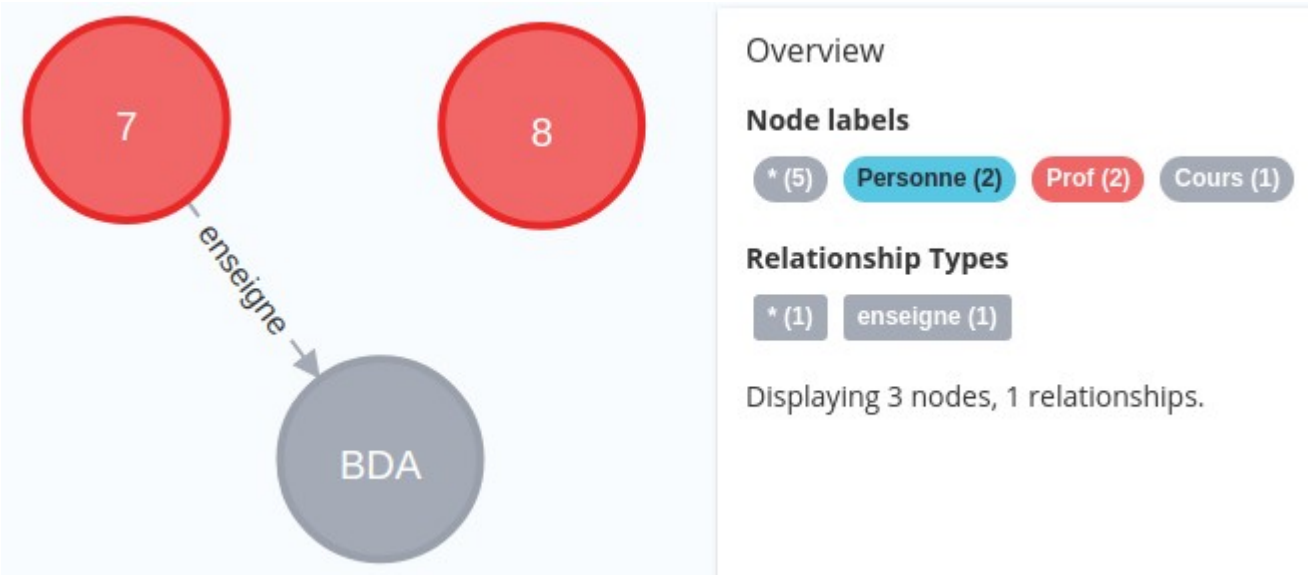
Représentation graphique

Cypher

```
create (n : Personne : Prof {nom : "Messai", statut : "MCF"})
create (n : Cours {titre : "BDA"})
match (n) return n
```

Concept 4 : Relation

Représentation graphique



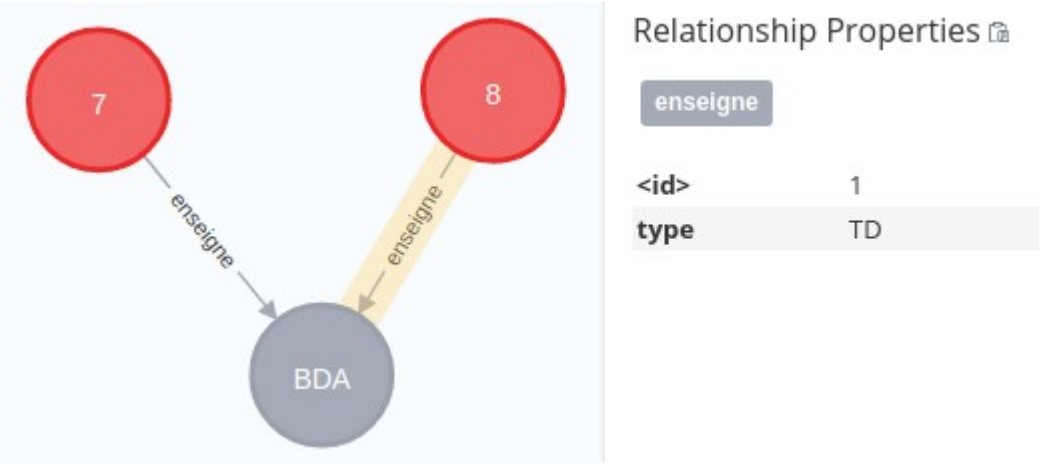
Cypher

```
match (p : Prof), (c : Cours)
where c.titre = "BDA" and p.nom = "Darmont"
create (p)-[:enseigne]->(c)
```

```
match (n) return n
```

Propriétés de relation

Représentation graphique



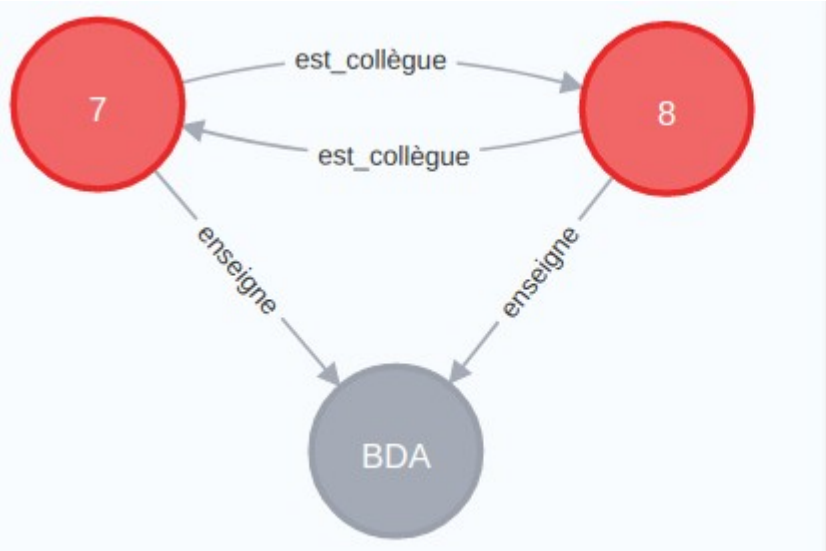
Cypher

```
match (p : Prof), (c : Cours)
where c.titre = "BDA" and p.nom = "Messai"
create (p)-[:enseigne {type : "TD"}]->(c)
```

```
match (n) return n
```

Relation bilatérales (1/2)

Représentation graphique



Overview

Node labels

* (5) | Personne (2) | Prof (2) | Cours (1)

Relationship Types

* (4) | est_collègue (2) | enseigne (2)

Displaying 3 nodes, 4 relationships.

Relation bilatérales (2/2)

Cypher

```
match (p1 : Prof), (p2 : Prof)
where p1.nom = "Darmont" and p2.nom = "Messai"
create (p1)-[:est_collègue]->(p2)
```

```
match (p1 : Prof), (p2 : Prof)
where p1.nom = "Messai" and p2.nom = "Darmont"
create (p1)-[:est_collègue]->(p2)
```

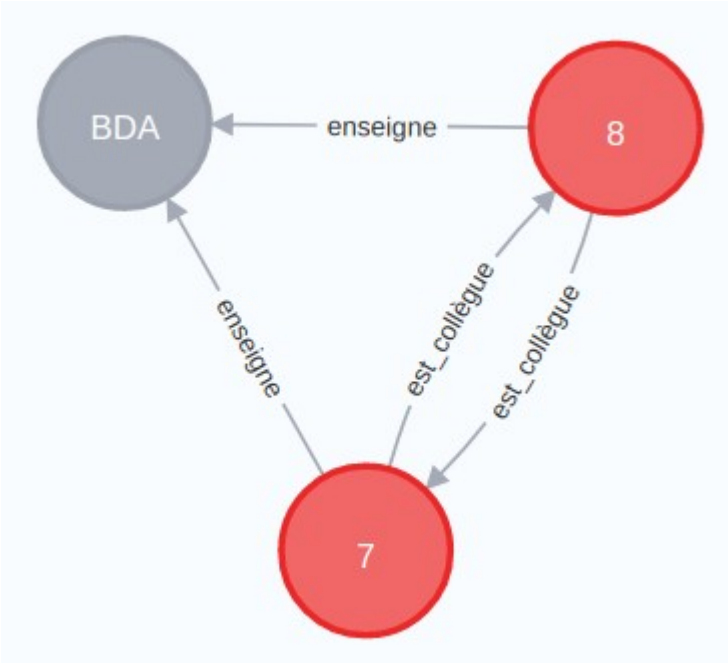
```
match (n) return n
```

Interrogation : tous les nœuds

Requête Cypher

```
match (n)
return n
```

Résultat

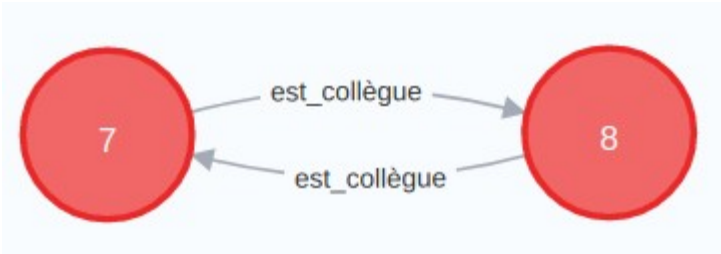


Interrogation : limite de taille

Requête Cypher

```
match (n)
return n
limit 2
```

Résultat



Interrogation : accès aux propriétés

Requête Cypher

```
match (p : Prof)
return p.nom
```

Résultat

	p.nom
1	"Darmont"
2	"Messai"

Interrogation : fonctions

Requête Cypher

```
match (p : Prof)
return toUpper(p.nom)
```

Résultat

	toUpper(p.nom)
1	"DARMONT"
2	"MESSAI"

https://neo4j.com/docs/cypher-manual/current/functions/

Interrogation : tri sur propriété(s)

Requête Cypher

```
match (p : Prof)
return p.nom
order by p.nom desc
```

Résultat

	p.nom
1	"Messai"
2	"Darmont"

Interrogation : agrégation sur propriété

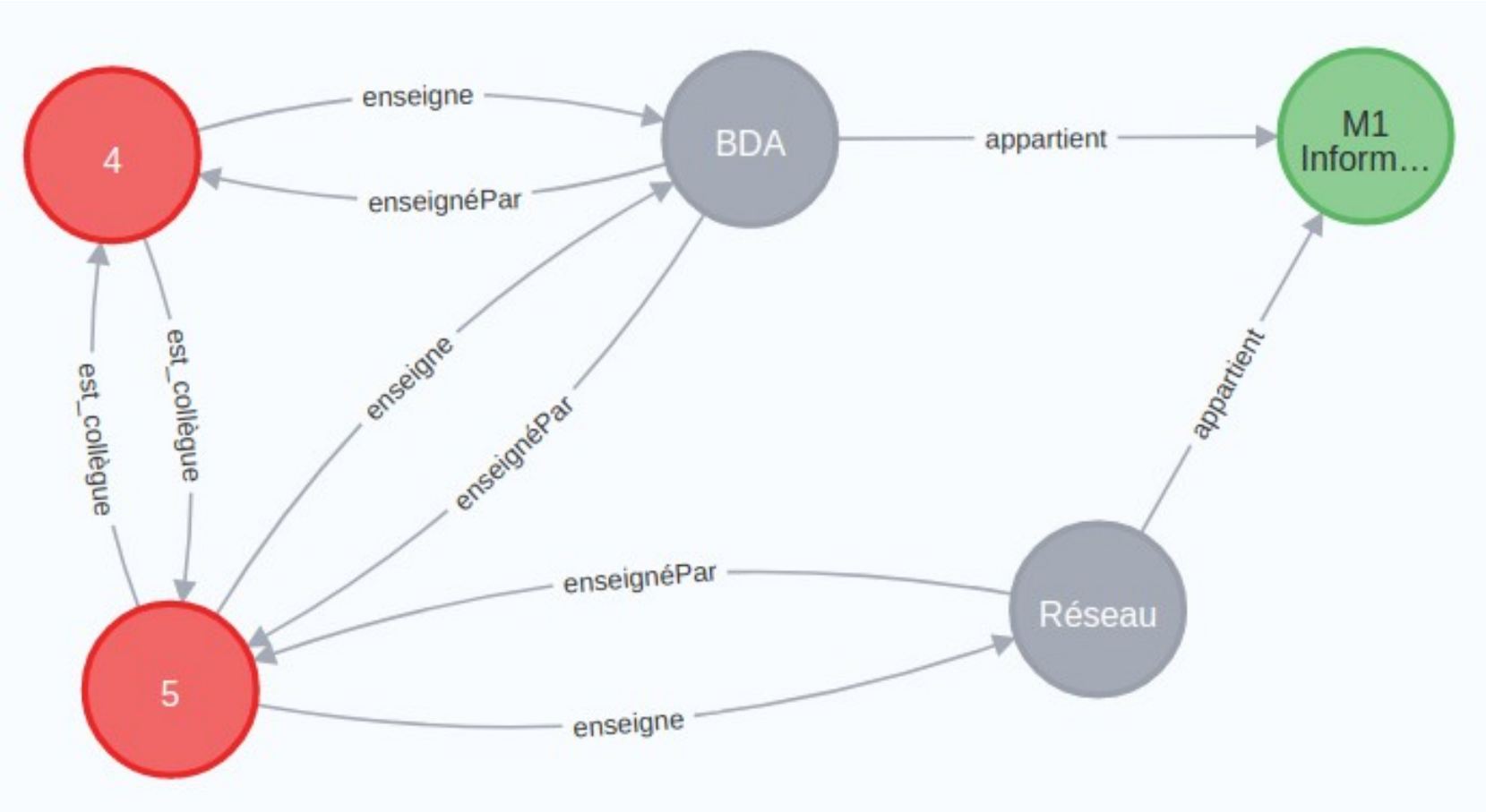
Requête Cypher

```
match (p : Prof)
return count(p)
```

Résultat

	count(p)
1	2

Étendons encore le graphe !



Gestion des doublons

Requête Cypher

1
```
match(c : Cours)-[:enseignéPar]-(p : Prof)
return p.nom
```

2
```
match(c : Cours)-[:enseignéPar]-(p : Prof)
return distinct p.nom
```

Résultat

1

	p.nom
1	"Darmont"
2	"Messai"
3	"Messai"

2

	p.nom
1	"Darmont"
2	"Messai"

Interrogation : groupement sur propriété (1/2)

Requête Cypher

```
match (p : Prof)-[:enseigne]->(c : Cours)
return c.titre, count(p)
```

Résultat

	c.titre	count(p)
1	"Réseau"	1
2	"BDA"	2

Interrogation : groupement sur propriété (2/2)

Requête Cypher

```
match (p : Prof)-[:enseigne]->(c : Cours)-[:appartient]->
                                                (d : Diplôme)
return d.nom, count(p)
```

Résultat

	d.nom	count(p)
1	"M1 Informatique"	3

Interrogation : groupements sur propriété

Requête Cypher

```
match (p : Prof)-[:enseigne]->(c : Cours)
with c.titre as t, count(p) as c
return max(c)
```

Résultat

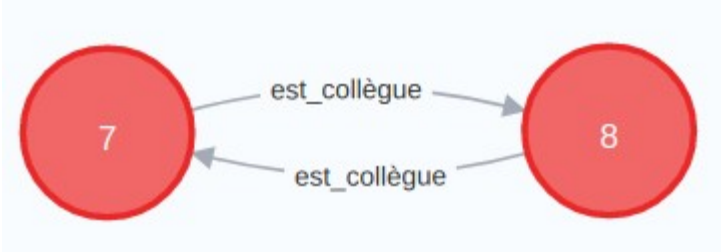
	max(c)
1	2

Interrogation : filtre sur label

Requête Cypher

match (n : Prof) return n	match (n : Personne : Prof) return n

Résultat

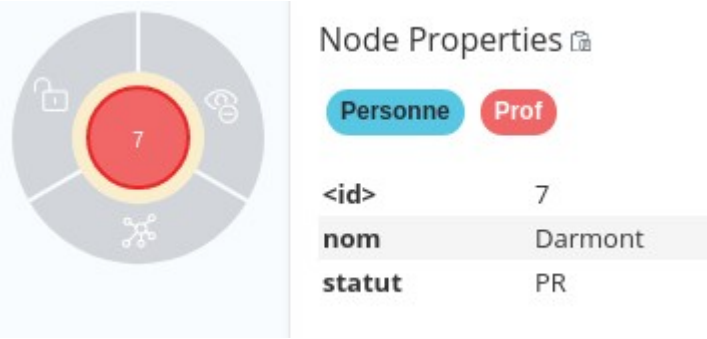


Interrogation : filtre sur propriété (1/2)

Requête Cypher

```
match (n : Prof {nom : "Darmont"})
return n
```

Résultat

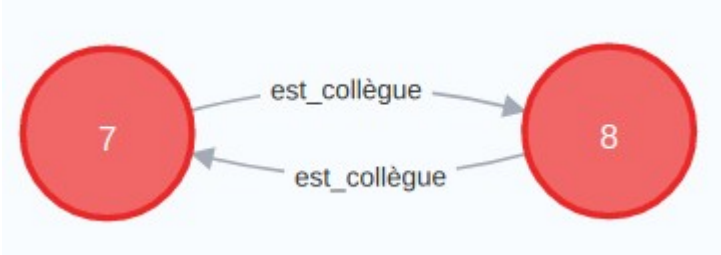


Interrogation : filtre sur propriétés (2/2)

Requête Cypher

```
match (n : Prof)
where n.nom = "Darmont" or n.statut = "MCF"
return n
```

Résultat

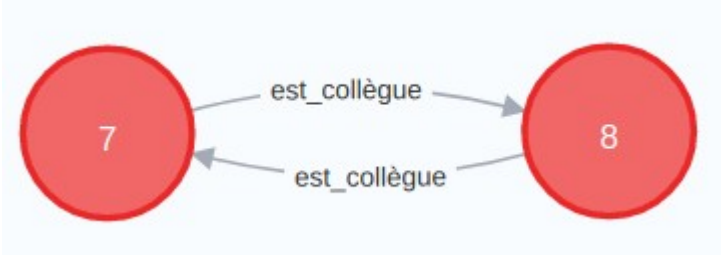


Interrogation : filtre sur relation

Requête Cypher

```
match (p : Prof)-[:enseigne]->(c : Cours)
where c.titre = "BDA"
return p
```

Résultat

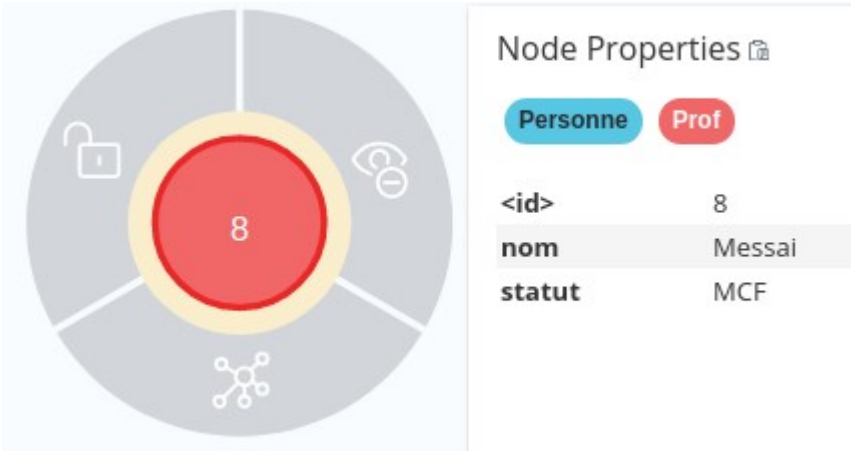


Interrogation : filtre sur propriétés de relation

Requête Cypher

```
match (p : Prof)-[e:enseigne]->(c : Cours)
where c.titre = "BDA" and e.type = "TD"
return p
```

Résultat



Index et contraintes d'unicité

Création d'index

```
create index on :Prof(nom)
```

Suppression d'index

```
drop index on :Prof(nom)
```

Création de contrainte d'unicité

```
create constraint on (c : Cours) assert c.titre is unique
```

Suppression de contrainte d'unicité

```
drop constraint on (c : Cours) assert c.titre is unique
```

Suppression de nœuds

Suppression simple

```
match (c : Cours {titre : "BDA"})
delete c
```

Suppression multiple

```
match (p1 : Prof {Name : "Darmont"}),
      (p2 : Personne {Name : "Messai"})
delete p1, p2
```

Suppression totale

```
match (n)
delete n
```

Ne fonctionne pas si des relations sont reliées au(x) nœuds.

Suppression de relations

Suppression ciblée

```
match (:Prof {nom : "Messai"})-[r:enseigne]-(:Cours {titre : "BDA"})
delete r
```

Suppression sur relation

```
match (:Prof)-[r:enseigne]-(:Cours)
delete r
```

Suppression sur type de relation

```
match ()-[r:est_collègue]-()
delete r
```

Suppression de la base de données entière

```
match (n) detach
delete n
```

Plan (partie 2)

- ✓ Objectifs
- ✓ Gestion de graphes avec Neo4J
- Interrogation de données JSON avec RumbleDB
- Autres outils

Quand utiliser un SGBD orienté documents ?

Données semi-structurées
Schéma flexible
Passage à l’échelle
Temps réel

Pourquoi ne pas utiliser MongoDB ?

Langage de requête limité (*template-based*)

Filtres de type and/or impossibles
Filtres sur des attributs différents impossibles
Jointures impossibles
Requêtes d'agrégation peu lisibles

https://www.linkedin.com/pulse/query-language-design-fail-jsoniq-vs-mongodb-pavel-velikhov/

RumbleDB : généralités

SGBD orienté documents en ligne de commande

Formats : JSON, Text, Parquet, CSV, SVM, ROOT...

Systèmes de fichiers : local, HDFS, S3, Azure, Spark...

Version 1.21.0 (2023)

Langage de requête fonctionnel basé sur XQuery

Types de données JSON

arborescentes	**n-uplets**
~ XML	~ relationnel étendu
Documents bien formés	Fragments de docs
Requêtes mal prises en charge avec RumbleDB	Requêtes faciles

Données JSON (1/2)

```
// stores.json
{ "store number" : 1, "state" : "CA" }
{ "store number" : 2, "state" : "CA" }
{ "store number" : 3, "state" : "MA" }
{ "store number" : 4, "state" : "MA" }
```

```
// products.json
{ "name" : "broiler", "category" : "kitchen", "price" : 100, "cost" : 70 }
{ "name" : "toaster", "category" : "kitchen", "price" : 30, "cost" : 10 }
{ "name" : "blender", "category" : "kitchen", "price" : 50, "cost" : 25 }
{ "name" : "socks", "category" : "clothes", "price" : 5, "cost" : 2 }
{ "name" : "shirt", "category" : "clothes", "price" : 10, "cost" : 3 }
```

Données JSON (2/2)

```
// sales.json
{ "product" : "broiler", "store number" : 1, "quantity" : 20  }
{ "product" : "toaster", "store number" : 2, "quantity" : 100 }
{ "product" : "toaster", "store number" : 2, "quantity" : 50 }
{ "product" : "toaster", "store number" : 3, "quantity" : 50 }
{ "product" : "blender","store number" : 3, "quantity" : 100 }
{ "product" : "blender", "store number" : 3, "quantity" : 150 }
{ "product" : "socks", "store number" : 1, "quantity" : 500 }
{ "product" : "socks", "store number" : 2, "quantity" : 10 }
{ "product" : "shirt", "store number" : 3, "quantity" : 10 }
```

Requêtes de projection (1/3)

Requête

```
for $p in json-file("products.json")          (: Je suis un commentaire :)
return $p
```

n-uplets : json-file() – arborescence : json-doc()

Résultat

```
{ "name" : "broiler", "category" : "kitchen", "price" : 100, "cost" : 70 }
{ "name" : "toaster", "category" : "kitchen", "price" : 30, "cost" : 10 }
{ "name" : "blender", "category" : "kitchen", "price" : 50, "cost" : 25 }
{ "name" : "socks", "category" : "clothes", "price" : 5, "cost" : 2 }
{ "name" : "shirt", "category" : "clothes", "price" : 10, "cost" : 3 }
```

Requêtes de projection (2/3)

Requête

```
for $p in json-file("products.json")
return { "nom" : $p.name }
```

(le format du retour est libre, mais de préférence en JSON)

Résultat

```
{ "nom" : "broiler" }
{ "nom" : "toaster" }
{ "nom" : "blender" }
{ "nom" : "socks" }
{ "nom" : "shirt" }
```

Affichage des clés

Requête

```
for $s in json-file("sales.json")
return { "keys" : [keys($s)] }
```

(tableau)

Résultat

```
{ "keys" : [ "product", "store number", "quantity" ] }
{ "keys" : [ "product", "store number", "quantity" ] }
{ "keys" : [ "product", "store number", "quantity" ] }
{ "keys" : [ "product", "store number", "quantity" ] }
{ "keys" : [ "product", "store number", "quantity" ] }
{ "keys" : [ "product", "store number", "quantity" ] }
{ "keys" : [ "product", "store number", "quantity" ] }
{ "keys" : [ "product", "store number", "quantity" ] }
{ "keys" : [ "product", "store number", "quantity" ] }
```

(il y a matière à optimisation !)

Requêtes de restriction (1/2)

Requête

```
for $s in json-file("stores.json")
where $s.state eq "MA"
return { "store number" : $s.''store number'' }
```

Résultat

```
{ "store number" : 3 }
{ "store number" : 4 }
```

Requêtes de restriction (2/2)

Requête

```
for $p in json-file("products.json")
where $p.category eq "kitchen" and $p.price ge 50
return { "name" : $p.name }
```

(opérateurs de comparaison : < lt, ≤ le, = eq, ≠ ne, ≥ ge, > gt)

Résultat

```
{ "name" : "broiler" }
{ "name" : "blender" }
```

Requêtes sur tableaux (1/2)

```
// products2.json
{ "name" : "broiler", "category" : ["kitchen"] }
{ "name" : "toaster", "category" : ["kitchen"] }
{ "name" : "blender", "category" : ["kitchen", "bar"] }
```

Requête

```
for $p in json-file("products2.json")
return { $p.name : $p.category[ ] }
```

Résultat

```
{ "broiler" : "kitchen" }
{ "toaster" : "kitchen" }
{ "blender" : [ "kitchen", "bar" ] }
```

Requêtes sur tableaux (2/2)

Requête

```
for $p in json-file("products2.json")
where $p.name eq "blender"
return $p.category[ ]
```

Résultat

```
"kitchen"
"bar"
```

Tri (1/2)

Requête

```
for $p in json-file("products.json")
order by $p.name
return { "name" : $p.name }
```

Résultat

```
{ "name" : "blender" }
{ "name" : "broiler" }
{ "name" : "shirt" }
{ "name" : "socks" }
{ "name" : "toaster" }
```

Tri (2/2)

Requête

```
for $p in json-file("products.json")
order by $p.category, $p.price descending
return {
      "name" : $p.name,
      "category" : $p.category,
      "price" : $p.price }
```

Résultat

```
{ "name" : "shirt", "category" : "clothes", "price" : 10 }
{ "name" : "socks", "category" : "clothes", "price" : 5 }
{ "name" : "broiler", "category" : "kitchen", "price" : 100 }
{ "name" : "blender", "category" : "kitchen", "price" : 50 }
{ "name" : "toaster", "category" : "kitchen", "price" : 30 }
```

Fonctions JSONiq

Requête

```
for $p in json-file("products.json")
let $nameUp := upper-case($p.name)
return { "name" : $nameUp }
```

Résultat

```
{ "name" : "BROILER" }
{ "name" : "TOASTER" }
{ "name" : "BLENDER" }
{ "name" : "SOCKS" }
{ "name" : "SHIRT" }
```

Toutes les fonctions :
https://rumble.readthedocs.io/en/latest/Function%20library/
not() est une fonction !

Requêtes de calcul en ligne

Requête

```
for $p in json-file("products.json")
let $margin := $p.price - $p.cost
return {
      "name" : $p.name,
      "margin" : $margin }
```

Résultat

```
{ "name" : "broiler", "margin" : 30 }
{ "name" : "toaster", "margin" : 20 }
{ "name" : "blender", "margin" : 25 }
{ "name" : "socks", "margin" : 3 }
{ "name" : "shirt", "margin" : 7 }
```

Requêtes d'agrégation

Requête

```
let $avgCost := avg(
      for $p in json-file("products.json")
      return $p.cost )
return { "Average cost" : $avgCost }
```

Résultat

```
{ "Average cost" : 22 }
```

Requêtes de groupement

Requête

```
for $s in json-file("sales.json")
let $pname := $s.product
group by $pname
return { $pname : sum($s.quantity) }
```

Résultat

```
{ "toaster" : 200 }
{ "shirt" : 10 }
{ "broiler" : 20 }
{ "socks" : 510 }
{ "blender" : 250 }
```

Requêtes de jointure

Requête

```
for  $p in json-file("products.json"), $s in json-file("sales.json")
where $p.name = $s.product
let $benefit := ($p.price - $p.cost) * $s.quantity
return { "product" : $p.name, "benefit" : $benefit }
```

Résultat

```
{ "product" : "broiler", "benefit" : 600 }
{ "product" : "toaster", "benefit" : 2000 }
{ "product" : "toaster", "benefit" : 1000 }
{ "product" : "toaster", "benefit" : 1000 }
{ "product" : "blender", "benefit" : 2500 }
{ "product" : "blender", "benefit" : 3750 }
{ "product" : "socks", "benefit" : 1500 }
{ "product" : "socks", "benefit" : 30 }
{ "product" : "shirt", "benefit" : 70 }
```

Jointures et groupement multiple

Requête

```
for $st in json-file("stores.json"), $sa in json-file("sales.json"),
    $p in json-file("products.json")
where $st."store number" = $sa."store number"
          and $sa.product = $p.name
let $state := $st.state, $category := $p.category
group by $state, $category
return { "state" : $state, "category" : $category,
             "revenue" : sum($p.price) }
```

Résultat

```
{ "state" : "MA", "category" : "clothes", "revenue" : 10 }
{ "state" : "CA", "category" : "clothes", "revenue" : 10 }
{ "state" : "CA", "category" : "kitchen", "revenue" : 160 }
{ "state" : "MA", "category" : "kitchen", "revenue" : 130 }
```

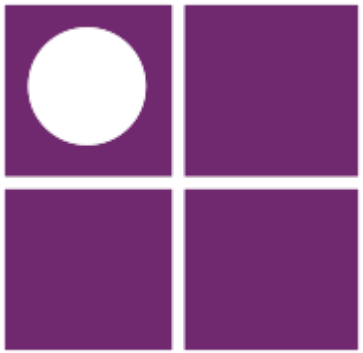
Plan (partie 2)

- ✓ Objectifs
- ✓ Gestion de graphes avec Neo4J
- ✓ Interrogation de données JSON avec RumbleDB
- Autres SGBD NoSQL

SGBD clés-valeurs

VMWare

Microsoft



Amazon SimpleDB

SGBD orientés colonnes

- Table Produit

ID	Désignation	Prix
205	Gear	75
221	Big bolt	650
285	Wheel belt	350

- Stockage physique en lignes
  - 205, Gear, 75 ; 221, Big bolt, 650 ; 285, Wheel belt, 350
- Stockage physique en colonnes
  - 205, 221, 285 ; Gear, Big Bolt, Wheel belt ; 75, 650, 350

SGBD orientés colonnes

- Recherche par prix
  - 205, Gear, 75 ; 221, Big bolt, 650 ; 285, Wheel belt, 350
  - 205, 221, 285 ; Gear, Big Bolt, Wheel belt ; 75, 650, 350
  - Colonnes : localité de référence (*clustering* physique)

Google Bigtable

SGBD orientés documents

MongoDB

amazon DynamoDB

Facebook → Apache

Couchbase

Apache

SGBD orientés graphes

Apache

Twitter

Plan (partie 2)

- ✓ Objectifs
- ✓ Gestion de graph 4J
- ✓ Interrogation de d N avec RumbleDB
- ✓ Autres SGBD NoSQL