Vers un modèle unifié de données entreposées et de données ouvertes liées

Concepts et expérimentations

Franck Ravat[1], **Jiefu Song**[1], **Olivier Teste**[2]

1. *IRIT - Université Toulouse I Capitole*
   2 rue du Doyen Gabriel Marty, 31042 Toulouse Cedex 09, France
   {ravat|song}@irit.fr

2. *IRIT - Université Toulouse II Jean Jaurès*
   1 place Georges Brassens, 31703 Blagnac Cedex, France
   teste@irit.fr

RÉSUMÉ. *De nos jours, la plupart des systèmes d'aide à la décision (SAD) reposent sur un entrepôt de données (ED) construit à partir de données de production internes à l'organisation. Cependant, les analyses décisionnelles peuvent être sensiblement améliorées par l'ajout d'informations supplémentaires provenant de l'extérieur d'une organisation, notamment des données ouvertes liées (DOL). L'intégration de ces données dans un SAD peut offrir de nouveaux points de vue aux décideurs. Dans cet article, nous décrivons un nouveau modèle multidimensionnel, appelé Cube Unifié, qui offre une représentation conceptuelle générique des données entreposées et des DOL. Un processus en deux étapes est proposé pour construire un Cube Unifié. Dans un premier temps, les schémas publiés avec des langages de modélisation spécifiques sont transformés en une représentation conceptuelle reposant sur un même langage. La seconde étape consiste à associer les schémas précédemment définis pour former un schéma unifié. Un langage algébrique est proposé afin de permettre aux concepteurs de construire un Cube Unifié selon leurs besoins. Pour valider nos propositions, nous montrons comment un Cube Unifié 1) est construit sur des jeux de données réelles et 2) permet aux décideurs d'effectuer des analyses décisionnelles avec de multiples sources.*

ABSTRACT. *Nowadays, most Decision Support Systems (DSS) rely on a Data Warehouse (DW) built with internal data sources. However, business analyses can be significantly enhanced by including complementary information coming from the outside of an organization, especially Linked Open Data (LOD). Integrating these data in a DSS can offer multiple perspectives to decision-makers. This paper provides a new multidimensional model, named Unified Cube, which offers a generic representation of both warehoused data and LOD at the conceptual level. A two-stage process is proposed to build a Unified Cube according to decision-makers' needs. As a first step, schemas published with specific modeling languages are transformed into a common conceptual representation based on a generic modeling language. The second step is to associate together all schemas obtained after the first step to form a Unified Cube containing all useful information about an analysis subject. A high-level declarative language*

is provided to enable designers to build a Unified Cube according to their needs. To demonstrate the feasibility of the proposed concepts, we show how a Unified Cube 1) is built from reel-world datasets and 2) enables decision-makers to carry out business analyses with multiple data sources.

MOTS-CLÉS : *données ouvertes liées, entrepôt de données, analyse multidimensionnelle.*

KEYWORDS: *linked open data, data warehouse, multidimensional analysis.*

DOI:10.3166/ISI.22.2.35-67

1. Introduction

Dans les systèmes d'aide à la décision (SAD), les données opérationnelles sont périodiquement extraites, transformées et chargées dans un environnement cohérent et centralisé, appelé entrepôt de données (ED), permettant d'organiser les données en cubes multidimensionnels. La prise de décision basée uniquement sur des données extraites des sources internes de l'organisation donne une vue partielle sur les activités de l'organisation. Dans le contexte actuel fortement concurrentiel, des informations supplémentaires provenant de l'extérieur d'une organisation doivent être intégrées dans un SAD. L'avantage d'intégrer des données externes comme les données issues de sites Web permet d'offrir de nouveaux points de vue aux décideurs (cf. Abelló *et al.*, 2015).

Intégrer des données externes dans les analyses décisionnelles nécessite de connaître la sémantique absolue de ces nouvelles données. Dans le domaine du Web sémantique, les données ouvertes liées[1] (DOL ou *Linked Open Data*) permettent de publier des données sémantiquement interconnectés et interprétables par des programmes informatiques. En accédant simplement aux fournisseurs de données, de nombreuses sources DOL multidimensionnelles peuvent être utilisées dans un contexte de prise de décision (cf. Zorrilla *et al.*, 2012). Toutefois, les données entreposées et les DOL sont stockées dans des espaces de stockage différents. Ainsi, au cours d'une analyse, les décideurs doivent « naviguer » au sein de plusieurs schémas pour collecter toutes les informations nécessaires aux prises de décision. La dispersion des données entreposées et des DOL conduit à des recherches répétitives d'informations sur différentes sources. En outre, les données entreposées et les DOL reposent sur des modèles de données différents et spécifiques à chaque domaine. Ainsi, effectuer des analyses faisant intervenir des données de types différents stockées sur plusieurs supports rend la tâche complexe et fastidieuse.

Dans ce contexte, notre objectif est d'offrir un environnement contenant toutes les données pertinentes pour les prises de décision. À cette fin, nous devons représenter des données entreposées et des DOL de manière unifiée et indépendante des langages de modélisation spécifiques. Pour faciliter la tâche d'analyse par des

1. http://linkeddata.org

décideurs, cette représentation unifiée ne doit reposer que sur des concepts orientés utilisateur.

Dans cet article, nous décrivons une solution de modélisation générique permettant d'unifier des données entreposées avec des DOL. Cette unification aboutit à un cube de données. Ce dernier repose sur 1) un schéma unifiant les schémas sources et 2) des alignements partiels des instances de sources hétérogènes. Tout d'abord, nous présentons un modèle multidimensionnel conceptuel, appelé *Cube Unifié*. Ce modèle est basé sur une représentation générique des données provenant de plusieurs sources pour un sujet d'analyse spécifique. Deuxièmement, en complément du modèle de *Cube Unifié*, nous proposons un processus de définition de schémas en deux étapes : 1) transformation de divers schémas en une représentation conceptuelle et générique et 2) association/combinaison des différentes données sources pour former un schéma unifié. A la fin du processus, les décideurs obtiennent un *Cube Unifié* contenant à la fois des données entreposées et des DOL. Pour valider la faisabilité des deux propositions, nous avons étudié comment un *Cube Unifié* peut être construit sur des jeux de données réelles.

Cet article est organisé comme suit. La section 2 présente notre cas d'étude et la section 3 les travaux liés à l'unification des données provenant de différentes sources. La section 4 présente le langage de modélisation multidimensionnelle au travers de ses concepts et de ses notations graphiques. La section 5 décrit le processus de construction d'un *Cube Unifié* à partir de sources multiples. Et la dernière section montre la faisabilité de notre solution *via* des évaluations expérimentales.

2. Cas d'étude

Dans une entreprise spécialisée dans la vente de climatisations, un décideur se réfère à un ED pour évaluer les chiffres d'affaire réalisés par les commerçants. L'ED est modélisé de manière multidimensionnelle : il contient un sujet d'analyse, nommé *SALES*, constitué d'un ensemble d'indicateurs numériques, à savoir *UNITPRICE, QUANTITY* et *REVENUE*. Chaque mesure peut être agrégée selon trois axes d'analyse : *SALESMAN*, *PRODUCT* et *DATE*. Cet ED est implanté à l'aide d'une base de données relationnelles. Chaque dimension est implantée via plusieurs tables (cf. figure 1). Par exemple, la dimension nommée *PRODUCT* est implantée avec quatre tables : *PRODUCT*, *BRAND*, *RANGE* et *SECTOR*. Chaque table représente un niveau de granularité. Les tables des niveaux inférieurs contiennent une ou plusieurs clés étrangères référençant les niveaux supérieurs. Par exemple, la table *PRODUCT* possède une clé primaire *P_Key* et deux clés étrangères référençant ses niveaux parents *BRAND* et *RANGE*. Le fait est implanté *via* une table de fait. L'ensemble de clés étrangères de la table de fait indique les niveaux de granularité les plus faibles des dimensions associées. Par exemple, la table de faits *SALES* comprend trois clés étrangères associées aux niveaux de granularité les plus bas (c'est-à-dire *P_KEY*, *S_KEY* et *D_KEY*) des trois dimensions.

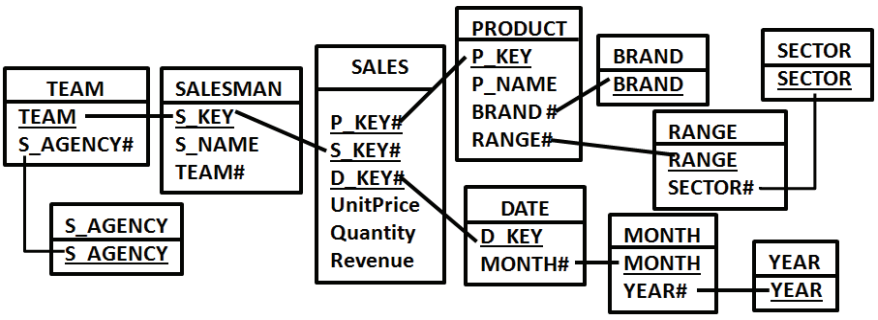


Figure 1. Implantation relationnelle de l'ED

L'ED seul ne fournit pas suffisamment d'informations pour une prise de décision efficace. Le décideur doit chercher des informations supplémentaires pour obtenir des perspectives complémentaires sur les activités de ventes. Pour mieux comprendre les relations entre les activités de ventes et les changements climatiques, le décideur consulte un ensemble de DOL en ligne qui révèle la température moyenne mensuelle selon les villes et les pays. Les DOL sont publiées dans le format *RDF Data Cube Vocabulary* (QB)[2], qui est un standard du W3C pour publier des données statistiques multidimensionnelles. Un schéma QB comprend un ensemble de `qb:Observations` [3] (proche de la définition des faits dans la terminologie de l'ED) qui décrit `qb:MeasureProperty` (mesures) avec `qb:DimensionProperty` (dimensions). Par exemple, dans la figure 2 le schéma `eg:CLIMATECHANGE` comprend deux instances de fait `qb:Observation` identifiées par `eg:ob1` et par `eg:ob2` (cf. lignes 15-24). Chaque instance de `qb:Observation` relie une instance de `eg:M_TEMPERATURE` avec une instance de chacune des dimensions, à savoir `eg:GEOGRAPHY` et `eg:TIME`.

Il est important de noter que, contrairement à la définition hiérarchique d'une dimension dans un ED, une dimension dans un schéma QB est un concept non hiérarchique. Le seul niveau de granularité d'une dimension de QB est représenté par `skos:hasTopConcept` au niveau du schéma. Par exemple, dans la figure 3 les attributs `eg:CITY` et `eg:COUNTRY` partagent le même niveau de granularité dans la dimension `eg:GEOGRAPHY` (cf. ligne 29).

Étant donné que les concurrents de la même zone de chalandise peuvent avoir une influence sur les ventes de la société, le décideur consulte aussi une source de DOL en ligne sur les prix de vente proposés par les concurrents. Cette DOL est publiée au format QB4OLAP ; elle décrit le prix d'un type de marchandises offert

2. http://www.w3.org/TR/vocab-data-cube
3. Les détails sur les préfixes sont disponibles sur http://prefix.cc/

par un fournisseur. De part la complexité syntaxique du vocabulaire, le décideur a des difficultés à explorer directement un schéma QB4OLAP (cf. figure 4).

```
eg:CLIMATE
    a qb:DataSet ;
    rdfs:label "Climate Changes"@en;
    qb:structure eg:CLIMCHANGE.
eg:CLIMCHANGE a qb:DataStructureDefinition;
    rdfs:label "Data Structure"@en;
    qb:component
        [ qb:dimension eg:TIME],
        [ qb:dimension eg:GEOGRAPHY];
    qb:component
        [qb:measureProperty eg:M_TEMPERATURE].
eg:M_TEMPERATURE a qb:MeasureProperty;
    rdfs:label "Temperature"@en;
    rdfs:range xsd:decimal.
eg:ob1  a qb:Observation;
   qb:dataSet eg:CLIMATE;
   eg:TIME eg:ym2015-11;
   eg:GEOGRAPHY eg:c1;
   eg:M_TEMPERATURE 12.
eg:ob2  a qb:Observation;
   qb:dataSet eg:CLIMATE;
   eg:TIME eg:ym2015-11;
   eg:GEOGRAPHY eg:co1;
   eg:M TEMPERATURE 10.
```

Figure 2. Extrait d'une collection de DOL publiées au format QB

```
eg:GEOGRAPHY a qb:DimensionProperty;
    rdfs:label "GEOGRAPHY"@en;
    qb:codeList eg:ATTSGEO.
eg:ATTSGEO a skos:ConceptScheme;
    skos:hasTopConcept eg:CITY, eg:COUNTRY.
eg:CITY a skos:Concept;
    rdfs:label "CITY"@en;
eg:COUNTRY a skos:Concept;
    rdfs:label "COUNTRY"@en.
eg:c1 a eg:CITY;
  rdfs:label "TOULOUSE"@en;
  skos:broader eg:co1.
eg:co1 a eg:COUNTRY;
  rdfs:label "FRANCE"@en.
```

Figure 3. Dimension GEOGRAPHY dans le schéma QB

```
eg:RETAIL
    a qb:DataSet ;
    rdfs:label "Outlet"@en ;
    qb:structure eg:RETPRICE.
eg:RETPRICE a qb:DataStructureDefinition;
    rdfs:label "Retail Price Dataset Structure"@en;
    qb:component
        [ qb4o:level eg:SHOP; qb4o:cardinality qb4o:ManyToOne],
        [ qb4o:level eg:TYPE; qb4o:cardinality qb4o:ManyToOne];
    qb:component
        [qb:measure eg:M_RETAILPRICE; qb4o:AggregateFunction
            qb4o:sum, qb4o:avg, qb4o:min, qb4o:max ].
eg:M_RETAILPRICE a qb:MeasureProperty;
    rdfs:label "RetailPrice"@en;
    rdfs:range xsd:decimal.
eg:RETAILER a qb:DimensionProperty;
    rdfs:label "RETAILER"@en;
    qb4o:hasHierarchy eg:H_COM, eg:H_CA.
eg:H_COM a qb4o:HierarchyProperty;
    rdfs:label "COMPANY HIERARCHY"@en;
    qb4o:inDimension eg:RETAILER;
    qb4o:hasLevel eg:L_SHOP, eg:L_COMPANY.
eg:H_CA a qb4o:HierarchyProperty;
    rdfs:label "CATCHMENTAREA HIERARCHY"@en;
    qb4o:inDimension eg:RETAILER;
    qb4o:hasLevel eg:L_SHOP, eg:L_CATCHMENTAREA.
eg:L_SHOP a qb4o:LevelInHierarchy ;
    qb4o:levelComponent eg:SHOP;
    qb4o:hierarchyComponent eg:H_COM, eg:H_CA.
eg:SHOP a qb4o:LevelProperty;
    rdfs:label "SHOP"@en.
eg:L_CATCHMENTAREA a qb4o:LevelInHierarchy ;
    qb4o:levelComponent eg:CATCHMENTAREA;
    qb4o:hierarchyComponent eg:H_CA.
eg:L_COMPANY a qb4o:LevelInHierarchy ;
    qb4o:levelComponent eg:COMPANY;
    qb4o:hierarchyComponent eg:H_COM.
eg:Step_l_sTol_ca a qb4o:HierarchyStep;
    qb4o:childLevel eg:L_SHOP;
    qb4o:parentLevel eg:L_CATCHMENTAREA;
    qb4o:cardinality qb4o:oneToMany.
eg:Step_l_caTol_co a qb4o:HierarchyStep;
    qb4o:childLevel eg:L_SHOP;
    qb4o:parentLevel eg:L_CATCHMENTAREA;
    qb4o:cardinality qb4o:oneToMany.
```

Figure 4. Extrait d'une collection de DOL publiées au format QB4OLAP

Les données entreposées et les DOL multidimensionnelles suivent différents modèles définis par des langages de modélisation spécifiques (cf. Zorrilla *et al.*, 2012 ; Bauer et Kaltenböck, 2012). La syntaxe complexe de ces langages de

modélisation complexifie la tâche d'analyse par des décideurs. De plus, comme chacun des schémas fournit une vue partielle du sujet d'analyse, le décideur doit naviguer parmi plusieurs schémas pour obtenir une vision complète de l'analyse. Confronté à ces problèmes, le décideur veut construire un schéma conceptuel unifié qui englobe à la fois les données de l'ED ainsi que les DOL QB et QB4OLAP. Dans la suite de cet article, nous présentons comment un schéma unifié peut être conçu et développé.

3. État de l'art

Dans la littérature scientifique, nous pouvons trouver de nombreux états de l'art, tels que (Abelló *et al.,* 2015 ; Laborie *et al.*, 2015), qui proposent d'intégrer à la fois des données entreposées et des DOL dans un schéma unifié. Ces états de l'art, mettent en évidence un consensus sur le besoin d'une solution de modélisation générique unifiant les données provenant de différentes sources.

Construire un schéma unifié à partir de multiples sources nécessite d'identifier des correspondances entre les schémas d'ED et de DOL. Pour ce faire, nous pouvons nous référer aux techniques d'*alignement de schémas*. Les auteurs de (Rahm et Bernstein, 2001) étudient différentes techniques d'alignement de schémas utilisées dans plusieurs domaines. En ce qui concerne l'unification de données entreposées et de DOL, l'alignement de schéma peut être une première réponse (cf. Bernstein *et al.,* 2011), notamment des techniques proposées dans le domaine de l'alignement d'ontologies (cf. Euzenat, 2013). Cependant, contrairement à une ontologie dont la sémantique est toujours explicitement présentée, un ED perd après son implantation la sémantique définie lors de la conception (cf. Euzenat, 2013). Appliquer directement les techniques d'alignement d'ontologies à des données entreposées n'est donc par recommandé. Des études doivent être effectuées afin d'analyser la faisabilité d'associer des données entreposées et des DOL *via* l'alignement d'ontologies. Pour nos besoins, nous proposons d'appliquer des techniques d'alignement d'ontologies suffisamment génériques pour être applicables aussi bien à des données entreposées qu'à des DOL. Les techniques basées sur des mesures de similarité de chaines de caractères satisferont nos besoins, étant donné qu'elles ne reposent pas sur la sémantique des données.

Utiliser des mesures de similarité permet uniquement d'établir les correspondances entre les composants de deux schémas. Des solutions doivent aussi être proposées pour la modélisation générique qui unifie des données de diverses sources. Les travaux existants relatifs à une telle modélisation peuvent être classés en trois approches.

Tout d'abord, la communauté des ED propose de construire un schéma multidimensionnel à partir des DOL en ligne. Plus précisément, des DOL sont considérées comme des données externes qui doivent être matérialisées dans un ED via un processus *Extract*, *Transform* and *Load* (ETL). Par exemple, les auteurs de

(Skoutas et Simitsis, 2007) proposent un *framework* ETL basé sur des ontologies pour intégrer des données issues du Web dans un ED. Ils se servent uniquement de la structure ontologique sans tenir compte des instances associées. Les auteurs de (Nebot *et al.*, 2009 ; Romero et Abelló, 2007) proposent de nouveaux processus ETL qui prennent en compte à la fois la structure et les instances d'une ontologie pour construire un ED. Les étapes nécessaires à un tel processus ETL sont décrites dans (Bellatreche *et al.,* 2013). Les auteurs de (Deb Nath *et al.,* 2015 ; Thenmozhi et Vivekanandan, 2014) montrent qu'un processus ETL basé sur des ontologies permet de réduire dans une certaine mesure le coût des mises à jour des données dans l'ED. Cependant, de par la lourdeur des traitements ETL et la volumétrie importante des DOL, il est toutefois difficile de maintenir la fraîcheur des DOL stockées dans un ED (cf. Etcheverry et Vaisman, 2012). Ainsi, un décideur peut difficilement obtenir des informations à jour lors de ses analyses.

Dans un second temps, la communauté des DOL vise à transformer et à représenter les données entreposées à l'aide d'un graphe RDF (cf. Saad *et al.,* 2013). Les schémas sont modélisés selon une structure multidimensionnelle avec des vocabulaires tels que *RDF Data Cube Vocabulary* (QB) ou ses extensions (cf. Kämpgen et Harth, 2011 ; Etcheverry et Vaisman, 2012). Les auteurs de (Etcheverry *et al.,* 2014) proposent le vocabulaire QB4OLAP en incluant des composants spécifiques aux modèles multidimensionnels, par exemple les niveaux de granularité multiples (`qb4o:LevelInHierarchy`) au sein de plusieurs chemins d'agrégation (`qb4o:HierarchyProperty`) et la spécification des fonctions d'agrégation (`qb4o:AggregateFunction`) associées à une mesure. Cependant, ces vocabulaires n'incluent aucun concept orienté métier. Il est donc difficile pour un utilisateur non-expert d'interagir directement avec ces vocabulaires.

Troisièmement, un effort conjoint entre les deux communautés consiste à combiner plusieurs sources d'un ED et des DOL. Les auteurs de (Abelló *et al.,* 2013) discutent des défis et des perspectives de recherche pour fusionner les données à partir de sources multiples, mais des propositions concrètes manquent. Les travaux de (Matei *et al.,* 2015) proposent le vocabulaire appelé IGOLAP qui permet de représenter à la fois les données entreposées et des DOL. Cependant, un schéma IGOLAP exige la transformation des données entreposées en un graphe RDF, qui subit les mêmes inconvénients que les travaux basés sur les processus ETL. De plus, un schéma IGOLAP se situe au niveau logique et donc pas orienté utilisateur.

Face à ces problèmes, notre solution de modélisation unifiée doit être suffisamment générique et indépendante des solutions de modélisation spécifiques aux données entreposées et aux DOL. De plus, elle doit permettre d'effectuer des analyses à la volée aussi bien sur des données entreposées que sur des DOL. Pour faciliter la tâche d'analyse effectuée par des décideurs, seuls les concepts orientés utilisateur doivent être utilisés.

4. Modélisation conceptuelle de *Cubes Unifiés*

Outre les différences dans la syntaxe des langages de modélisation, les communautés des ED et des DOL ne partagent pas les mêmes concepts : le domaine des ED et plus généralement des bases de données se concentre principalement sur la structure des données (c'est-à-dire le schéma). Le domaine des DOL repose sur des vocabulaires standardisés et des ontologies de domaines pour publier des instances de données interconnectées. Afin de représenter à la fois des données entreposées et des DOL, le *Cube Unifié* doit inclure des définitions relatives aux schémas et aux instances.

Dans cette section, nous décrivons un langage de modélisation multidimensionnelle permettant de définir un *Cube Unifié*. Ce langage de modélisation est suffisamment générique pour inclure dans un seul schéma à la fois des données entreposées et des DOL multidimensionnelles. Des notations graphiques sont également proposées pour le *Cube Unifié* afin de faciliter les analyses. Ce langage de modélisation est basé sur les concepts de dimensions et de cubes multidimensionnels. Chacun d'eux est défini par un schéma et possède ses propres instances.

4.1. Schéma de dimension

Un *schéma de dimension* représente la structure d'un axe d'analyse composé d'attributs organisés en un ou plusieurs niveaux d'agrégation. La définition de *schémas de dimension* doit être suffisamment générique pour pouvoir représenter les données provenant aussi bien des ED que des DOL contenant 1) de multiples attributs dans un niveau d'agrégation, 2) des dimensions mono-hiérarchiques, 3) des dimensions multi-hiérarchiques et, 4) des dimensions non hiérarchisées.

Définition 1. Un *schéma de dimension* est défini par $D_i=\langle dn^i, \mathcal{L}^i, \prec^i\rangle$, où :

– dn^i : nom de la dimension ;

– L^i : ensemble fini de couples de $\langle l_x, A_x\rangle$, tel que l_x est un niveau et A_x est un ensemble d'attributs associé au niveau l_x. Chaque attribut a_k ($a_k \in A_x$) est une paire $\langle an^k, \sigma^k\rangle$, telle que an^k est le nom de l'attribut, σ^k est une formule permettant d'extraire directement toutes les instances de l'attribut depuis la source. Le domaine d'un attribut est représenté par $dom(a_k)$;

– $\prec^i$: ensemble de relations binaires asymétriques et transitives qui traduisent une relation d'agrégation entre des niveaux.

REMARQUE.— Dans le cas d'une dimension non hiérarchique (par exemple dimension d'un schéma de QB), un seul niveau contient tous les attributs de la dimension.

Pour compléter la définition d'un *schéma de dimension*, nous proposons la notation graphique présentée en figure 5 pour représenter un *schéma de dimension*. Un rectangle contient le nom de la dimension. Chaque niveau est représenté par un

rond associé au nom d'un attribut caractérisant une granularité. Les attributs descriptifs sont liés à un niveau par un lien non directionnel. Une relation binaire est représentée par une flèche partant du niveau fils vers le niveau père.

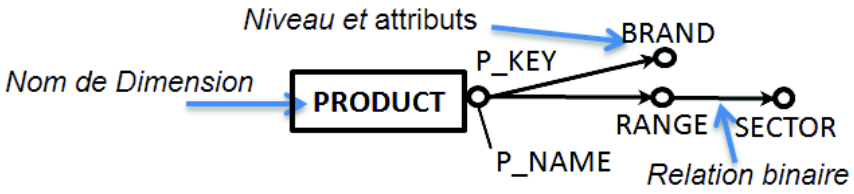


Figure 5. Notation graphique d'un schéma de dimension

Dans la suite de cet article, afin de simplifier la lecture des définitions, nous proposons d'utiliser $\mathcal{L}$ et $\prec$ au lieu de $\mathcal{L}^i$ et $\prec^i$ lorsque les formules font référence à une seule dimension D_i.

4.2. Instances de dimension

La définition des *instances de dimension* est nécessaire pour assurer l'interopérabilité entre les modèles centrés schéma (ED) et ceux centrés instances (DOL). Une *instance de dimension* (i) associe un attribut à ses instances, (ii) décrit les relations d'agrégation hiérarchiques entre les instances d'attributs.

Définition 2. Une *instance de dimension* est définie par $ID_j = \langle F^j, R^j \rangle$, où :

– F^j : ensemble de fonctions : chaque fonction notée f_k associe un attribut a_k ($a_k \in \mathcal{A}_x$) au niveau l_x ($l_x \in \mathcal{L}$) avec les instances de cet attribut (c'est-à-dire un ensemble de constantes dans $\bigcup_{l_x \in \mathcal{L}} \pi_{l_x}(D)$[4]) ;

– R^j : ensemble de fonctions du type *rollup* entre un attribut fils et un attribut père. Chaque fonction notée $\Uparrow_{a_x}^{a_y}$ associe l'attribut a_x avec l'attribut a_y, tel que $\Uparrow_{a_x}^{a_y}$: $dom(a_x) \rightarrow dom(a_y)$.

4.3. Schéma de Cube Unifié

Un *schéma de Cube Unifié* représente la structure multidimensionnelle de données provenant d'une ou plusieurs sources. Les dimensions provenant de sources différentes sont reliées entre elles, afin que les décideurs puissent obtenir de nouvelles perspectives d'analyse et des informations complémentaires lors des analyses.

4. π représente l'opérateur de projection dans l'algèbre relationnelle.

Définition 3. Un *schéma de Cube Unifié* est défini par S=⟨ucn, $\mathcal{D}$, $\mathcal{M}$, $\mathcal{RC}$⟩, où :

– ucn : nom du schéma de Cube Unifié ;

– $\mathcal{D}$={D_1;…;D_n} : ensemble fini de dimensions ;

– $\mathcal{M}$={m_1;…;m_k} : ensemble fini d'indicateurs numériques appelés *mesure*s ;

– $\mathcal{RC}$: ensemble de *relations corrélatives* décrivant les correspondances entre les données. Une *relation corrélative* est une application interdimension qui associe deux niveaux de deux dimensions différentes. Une *relation corrélative* possède un *niveau de départ* et un *niveau d'arrivée*.

REMARQUE.— La définition de schémas de *Cubes Unifié* est suffisamment générique pour représenter la structure des données provenant d'une ou plusieurs sources. Dans le cas où une seule source de données est impliquée, nous obtenons un *Cube Unifié* sans *relations corrélatives* ($\mathcal{RC} = \varnothing$). Dans les autres cas, les *relations corrélatives* doivent être construites pour représenter les relations entre les données de différentes sources. Plus de détails sur la *relation corrélative* sont disponibles en section 0.

La figure 6 présente la notation graphique d'un *Cube Unifié* construit uniquement sur les données entreposées. Un rectangle contient le nom d'un *Cube Unifié* et un ensemble de mesures. Les mesures partageant les mêmes dimensions sont regroupées afin de faciliter l'analyse.

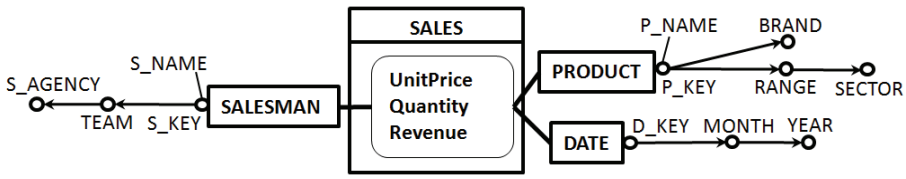


Figure 6. Notation graphique d'un Cube Unifié contenant des données entreposées

4.4. Instance d'un Cube Unifié

Cette définition permet : 1) d'associer des instances d'attributs de dimensions aux valeurs de mesures et 2) de relier les instances d'attributs *via* les *niveaux de départ* et *d'arrivée*.

Définition 4. Soit {a_1,..., a_n} un ensemble d'attributs, parmi lesquels a_x est un attribut de la dimension D_x, *dom*(m_j) est un ensemble de valeurs de la mesure m_j ($m_j \in \mathcal{M}$), une *instance de Cube Unifié* est définie par ICU_p =⟨ɱ, Ł⟩, où :

– ɱ : une fonction entre des valeurs de mesures et un ensemble d'instances d'attributs, tel que $\forall m_j \in \mathcal{M}$, ɱ : $dom(m_j) \rightarrow 2^{\cup_{x\in[1,n]} dom(a_x)}$;

– $\exists\ a_i \in \mathcal{A}^{départ}$, $\exists\ a_j \in \mathcal{A}^{arrivée}$, Ł : $dom(a_i) \rightarrow dom(a_j)$: une instance de *relation corrélative*. Elle associe un ensemble d'instances $dom(a_i)$ d'un attribut du *niveau de départ* à des instances *dom*(a_j) d'un attribut situé au *niveau d'arrivée*.

5. Processus de construction d'un *Cube Unifié*

Grâce à la modélisation conceptuelle de *Cubes Unifiés*, des données entreposées et des DOL peuvent être unifiées dans une représentation générique. Afin d'obtenir une telle représentation, nous présentons dans cette section, un processus de construction d'un *Cube Unifié* à partir de sources multiples.

Comme nous pouvons le voir dans la figure 7, la première étape transforme les schémas de données publiées avec des vocabulaires spécifiques en une représentation conceptuelle exprimée avec un langage de modélisation commun et indépendant des domaines de l'ED et des DOL. Le langage de modélisation multidimensionnelle de *Cube Unifié* est utilisé dans l'étape 1. La représentation obtenue après la première étape est appelée *cube d'exportation*, son but est de faciliter la combinaison de différentes sources dans un *Cube Unifié*.

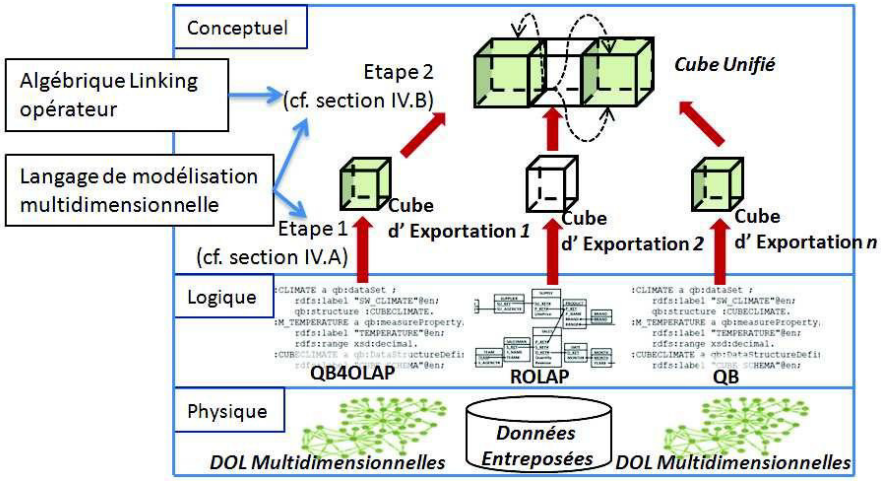


Figure 7. Construire un Cube Unifié à partir de sources de données multiples

La deuxième étape met en relation des données connexes qui sont réparties dans différents *cubes d'exportation*. Pour ce faire, le langage de modélisation multidimensionnelle *Cube Unifié* propose le concept de *relations corrélatives*, qui permet de représenter des correspondances entre instances d'attributs connexes. Pour faciliter la construction de *relations corrélatives*, nous proposons un langage algébrique permettant de relier deux collections d'instances d'attributs.

Le résultat du processus de construction est un *Cube Unifié* contenant toutes les données utiles pour les analyses décisionnelles.

5.1. Représentation Générique Conceptuelle

Les données entreposées et les DOL sont publiées selon différents modèles aux niveaux logique et physique. Chaque modèle de données repose sur un langage de modélisation spécifique. Afin d'avoir une représentation unifiée de l'ensemble des données nécessaires aux prises de décision, nous proposons de transformer différentes sources en une représentation générique à l'aide d'un langage de modélisation conceptuel commun.

Le langage de modélisation de *Cube Unifié* est suffisamment expressif pour couvrir les caractéristiques multidimensionnelles des schémas OLAP, QB et QB4OLAP. Un *cube d'exportation* exprimé avec le langage de modélisation *Cube Unifié* est automatiquement obtenu en se référant aux règles de traduction présentées ci-dessous.

5.1.1. D'un schéma OLAP à un cube d'exportation

La transformation d'un schéma OLAP (entrepôt de données représenté à l'aide de cubes multidimensionnels) dans une représentation conceptuelle a été amplement étudiée au cours de la dernière décennie (cf. Chaudhuri et Dayal, 1997). Etant donné que le langage de modélisation de *Cube Unifié* couvre toutes les caractéristiques multidimensionnelles des modèles OLAP, la transformation d'un schéma OLAP en un *cube d'exportation* est assez directe et n'a donc pas besoin d'être explicitée dans cet article. Les notations graphiques du *cube d'exportation* obtenu du R-OLAP ED (cf. figure 1) peuvent être trouvées dans la figure 6.

5.1.2. D'un schéma QB4OLAP à un cube d'exportation

Extension du vocabulaire QB, le vocabulaire QB4OLAP permet de représenter un schéma multidimensionnel à l'aide des triplets RDF. Plus précisément, QB4OLAP contient des classes et des propriétés (préfixées par qb4o) afin de représenter 1) un niveau d'agrégation avec plusieurs attributs (à savoir, un paramètre et ses attributs faibles), 2) des dimensions avec plusieurs niveaux d'agrégation et 3) l'ensemble des fonctions d'agrégation associées à une mesure. Le tableau 1 donne un aperçu des définitions conceptuelles des *cubes d'exportation* et des triplets QB4OLAP correspondants. Notez que dans le domaine des DOL, une variable commence par un point d'interrogation.

Tableau 1. Concepts de Cube Unifié et triplets QB4OLAP

Concepts de *Cube Unifié*	**Triplets QB4OLAP**
Schéma de Dimension (?dim)	?dim `a qb:DimensionProperty`;
Niveau (?lvl)	?dim `qb4o:hasHierarchy` ?hierarchy; ?hierarchy `a qb4o:hierarchyProperty`; ?hierarchy `qb4o:hasLevel` ?lvl.
Attributs (?para and ?att)	?lvl `a qb4o:LevelInHierarchy`; ?lvl `qb4o:levelComponent` ?para; ? para `a qb:LevelProperty`; ?para `qb4o:hasAttribute` ?att.
Relation binaire (?biRel)	?biRel `a qb4o:HierarchyStep`; ?biRel `qb4o:childLevel` ?lvlChild; ?biRel `qb4o:parentLevel` ?lvlParent; ?biRel `qb4o:cardinality` ?cardinalityParentChild
Instance de dimension (?dimIns, ?dimInsParent)	?dimIns `qb4o:inLevel` ?lvlChild; ?dimIns `rdfs:label` ?InsChild; ?dimIns `skos:broader` ?dimInsParent. ?dimInsParent `qb4o:inLevel` ?lvlParent; ?dimInsParent `rdfs:label` ?InsParent.
Schéma de *Cube Unifié* (?cube)	?cube `a qb:DataSet`; ?cube `rdfs:label` ?cubeName; ?cube `qb:structure` ?cubeStructure; ?cubeStructure `a qb:DataStructureDefinition`; ?cubeStructure `qb:component` `[qb4o:level` ?lvlRoot; `qb4o:cardinality` ?cardinalityCubeDim`]`; ?cubeStructure `qb:component [qb:measure` ?measure`]`.
Mesure (?measure)	?measure `a qb:MeasureProperty`; ?measure `rdfs:label` ?mName.
Instance de *Cube Unifié* (?ob)	?ob `a qb:Observation`; ?ob `qb:dataSet` ?cube; ?ob ?dim ?dimIns; ?ob ?measure ?value.

Après application des principes énoncés dans le tableau 1 aux DOL relatives aux ventes, nous obtenons le *cube d'exportation* suivant (figure 8).

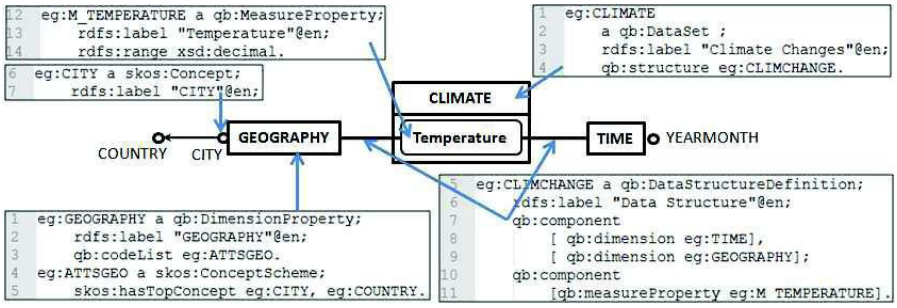


Figure 8. Cube d'exportation pour DOL au format QB4OLAP

5.1.3. D'un schéma QB à un cube d'exportation

QB est le standard actuel du W3C pour publier des DOL statistiques. Il permet de définir la structure d'un fait via *qb:DataStructureDefinition*. Contrairement à un modèle multidimensionnel dans lequel une dimension dispose de multiples niveaux de granularité, un fait QB est décrit par un ensemble de dimensions non-hiérarchisées. Afin de trouver la relation hiérarchique entre deux attributs dans une dimension de QB, nous pouvons nous référer aux instances de dimension. Cependant, l'obtention d'un schéma de dimension hiérarchique à partir d'un ensemble de données de QB n'est pas aisée. Par exemple, la propriété *skos:hasTopConcept* (ou *qb:hierarchyRoots* dans le cas d'une hiérarchie de type *non-SKOS*) est, par convention, utilisée pour relier un schéma de dimension à l'attribut le plus élevé dans une hiérarchie. Mais comme il n'y a aucune contrainte d'intégrité associée à cette propriété, un schéma de dimension compatible avec les spécifications de QB n'est pas nécessairement juste dans un contexte multidimensionnel comme le montre l'exemple ci-dessous.

```
eg:DimSchema skos:hasTopConcept eg:OneAttribute
eg:OneAttribute skos:broader eg:AnotherAttribute
eg:AnotherAttribute skos:inScheme eg:DimSchema
```

Afin d'obtenir automatiquement un *cube d'exportation* contenant des contraintes d'intégrité cohérentes (cf. Ghozzi *et al.*, 2005), nous proposons l'algorithme suivant.

Algorithme Construire un *cube d'exportation* à partir d'un schéma de QB

Entrée : *qbCube* est un schéma en QB où *cubeStructure* est la structure du schéma ; D_{QB} est l'ensemble des dimensions ; M_{QB} correspond à l'ensemble des mesures.

Sortie : Un *cube d'exportation* au niveau conceptuel.

Début

1. Pour chaque $d_i \in D_{QB}$ (d_i `a qb:DimensionProperty`)
2. Créer une dimension avec le nom *dimN* tel que d_i `rdfs:label` *dimN* ;
3. Pour chaque attribut a_k de la dimension d_i (d_i `qb:codeList` cl_i; cl_i `a skos:ConceptScheme`; cl_i `skos:hasTopConcept` a_k)
4. Créer un niveau l_k pour a_k;
5. Associer $\langle l_k, a_k \rangle$ à d_i;
6. Pour toutes instances I_k de l'attribut a_k (I_k `a` a_k)
7. Si I_k `skos:broader` I_{k+1} (I_{k+1} `a` a_{k+1})
8. Créer une *relation binaire* $l_k \prec l_{k+1}$;
9. Sinon Si I_k `skos:narrower` I_{k-1} (I_{k-1} `a` a_{k-1})
10. Créer une *relation binaire* $l_{k-1} \prec l_k$;
11. Fin si
12. Fin pour
13. Fin pour
14. Fin pour
15. Créer un *cube d'exportation* avec le nom *cubeN*, tel que *qbCube* `a qb:DataSet`; *qbCube* `rdfs:label` *cubeN*;
16. Pour chaque $m_j \in M_{QB}$ (m_j `a qb:MeasureProperty`)
17. Créer une mesure dans le *cube d'exportation* avec le nom *meN*, tel que m_j `rdfs:label` *meN*;

18. Associer cette mesure avec un ensemble de dimensions $D_{mj} \subseteq D_{QB}$, tel que $\forall d_x \in D_{mj}$:
cubeStructure `a qb:DataStructureDefinition;`
cubeStructure `qb:component [qb:dimension` d_x `],`
`qb:component [qb:measureProperty` m_j `];`
19. Fin pour
Fin.

REMARQUE.– Comme les auteurs de (Kämpgen et Harth, 2011) le mentionnent, le format QB n'est pas toujours correctement utilisé dans les cas réels d'application. Pour éviter d'apporter des informations inexactes aux décideurs, les méthodes de détection d'erreurs telles que celles présentées dans (Fleischhacker *et al.*, 2014) doivent être appliquées à un schéma QB avant d'appliquer l'algorithme proposé.

Nous appliquons l'algorithme pour l'ensemble des données QB sur les changements climatiques. Le *cube d'exportation* obtenu est représenté dans la figure 9. Après avoir transformé chaque source de données, nous regroupons ensemble tous les *cubes d'exportation* dans un *Cube Unifié* préliminaire. Ce *Cube Unifié* préliminaire vise à fournir une vue globale de toutes les données utiles pour les analyses.

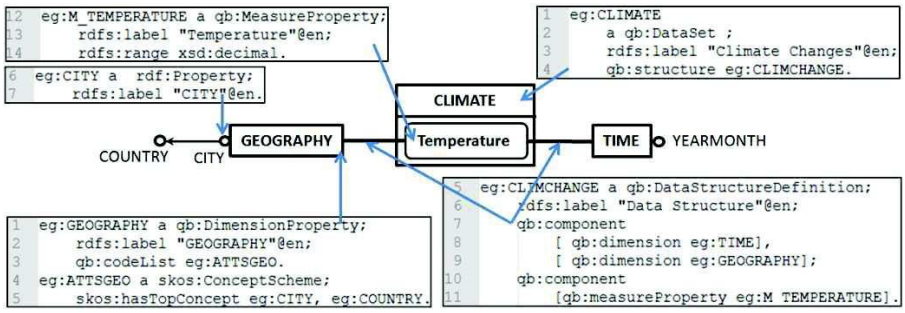


Figure 9. Cube d'exportation pour DOL au format QB

5.2. Relier les cubes d'exportation

Après avoir obtenu des *cubes d'exportation* à partir de multiples sources, la deuxième étape consiste à unifier l'ensemble des *cubes d'exportation*. Le langage de modélisation de *Cube Unifié* dispose des *relations corrélatives* permettant de représenter les correspondances entre des données entreposées et des DOL.

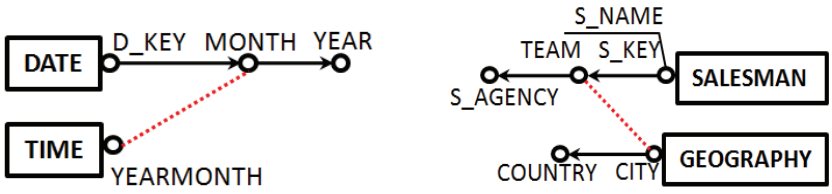
Dans cette section, nous discutons comment relier des *cubes d'exportation*. Dans un premier temps, nous décrivons la relation *corrélative*. Puis nous proposons un langage déclaratif présenté sous la forme d'un opérateur algébrique pour construire des *relations corrélatives*.

5.2.1. Relation corrélative

La relation entre les données provenant de sources multiples peut prendre différentes formes. Parmi les différents types de relations, la *relation corrélative* est particulièrement concernée dans le contexte de *Cubes Unifiés*.

Classiquement une granularité d'analyse correspond à un seul niveau de dimension. Dans le contexte de *Cube Unifié*, la notion de granularité d'analyse doit être généralisée, parce que plusieurs niveaux de dimension provenant de différentes sources peuvent se référer à la même granularité. De ce fait, les *relations corrélatives* doivent être établies entre deux niveaux de dimension référant au même concept. Ces deux niveaux peuvent être équivalents sémantiquement. Par exemple, les niveaux *MONTH* et *YEARMONTH* se réfèrent au même concept sur les dimensions *DATE* et *TIME* (cf. figure 10a). Dans la notation graphique, les *relations corrélatives* sont représentées par des pointillés.

Deux niveaux qui sont sémantiquement hétérogènes, mais qui ont le même rôle dans des analyses décisionnelles peuvent également être reliées par une relation corrélative. Par exemple, les dimensions *SALESMAN* et *GEOGRAPHY* sont deux axes d'analyse sémantiquement hétérogènes. Cependant, dans un contexte d'analyse spécifique où chaque *TEAM* de *SALESMAN* est en charge des ventes de chaque *CITY*, les analyses impliquant des *TEAM*s peuvent ainsi être effectuées avec des *CITIES*. Dans ce cas, les dimensions *SALESMAN* et *GEOGRAPHY* peuvent être considérées comme deux concepts analytiquement équivalents, qui doivent être associés ensemble par une *relation corrélative* (cf. figure 10b).



(a) Dimensions sémantiquement équivalentes (b) Dimensions analytiquement équivalentes

Figure 10. Relations corrélatives au niveau schéma

5.2.2. Construire une relation corrélative

Dans cette section, nous définissons un opérateur de liaison, nommée *DLink*, qui permet d'associer deux dimensions disjointes dans un *Cube Unifié*. La représentation algébrique de cet opérateur est présentée dans le tableau 2.

Nous utilisons le terme *d'alignement de dimensions* pour désigner l'opération de construction d'une *relation corrélative* entre deux dimensions via l'operateur *DLink*. Au niveau schéma, l'opérateur *DLink* associe deux dimensions avec une *relation corrélative*. Au niveau de l'instance, comme indiqué dans la figure 11, la *relation corrélative* correspond à une injection entre deux collections d'instances d'attributs.

L'injection implique qu'une instance d'attribut de départ est associée à au plus une instance d'arrivée. La *relation corrélative* dans la figure 11 est construite par l'opérateur suivant : ***DLink***(D_{DATE}; l_{Month}; D_{TIME}; $l_{YearMonth}$).

Tableau 2. Représentation algébrique

DLink ($D_{départ}$; $l_{départ}$; $D_{arrivée}$; $l_{arrivée}$)=*Cube Unifié*	
Entrée	- $D_{départ}$: dimension de départ de la *relation corrélative* ; - $l_{départ}$: niveau de départ de la *relation corrélative* ; - $D_{arrivée}$: dimension d'arrivée de la *relation corrélative* ; - $l_{arrivée}$: niveau d'arrivée de la *relation corrélative*.
Sortie	*Cube Unifié* : un *Cube Unifié* avec un ensemble actualisé de *relations corrélatives*.

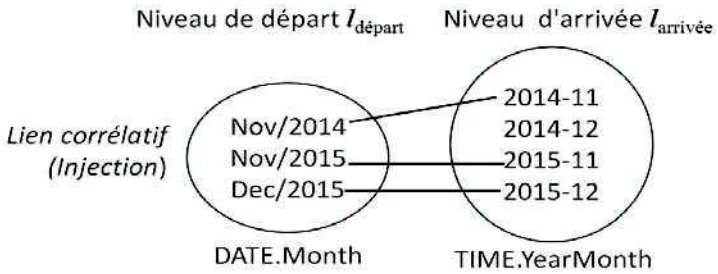


Figure 11. Relations Corrélatives au niveau instance

Selon le type de *relations corrélatives*, l'exécution de cet opérateur peut être semi-automatique ou automatique. Lorsque la relation entre deux dimensions dépend du contexte d'analyse, une approche semi-automatique doit être appliquée. Dans ce cas, le système propose aux décideurs plusieurs possibilités de correspondances entre les deux niveaux basés sur une ontologie de domaine, annotations sémantiques ou des contraintes d'intégrité (cf. Doan et Halevy, 2005 ; Cabanac *et al.*, 2009). Ensuite, les décideurs peuvent choisir ou modifier les correspondances proposées selon ses besoins.

Si les deux dimensions sont sémantiquement cohérentes, les opérations de liaison peuvent être automatisées. Notamment, nous pouvons nous référer aux approches présentées dans (Rahm, 2011) pour établir automatiquement ces correspondances. La section suivante de l'article se focalise sur les dimensions sémantiquement cohérentes. Nous allons étudier de l'efficience et de l'efficacité de l'alignement de dimensions *via* des expérimentations avec des données réelles.

6. Evaluations expérimentales sur l'alignement de dimensions

Une des étapes clés dans le processus de construction d'un *Cube Unifié* consiste à aligner des dimensions en corrélation. Pour ce faire, les *relations corrélatives* doivent être établies au niveau de l'instance de dimension. Comme la plupart des instances sont de type chaînes de caractères, les mesures de similarité jouent un rôle primordial dans le processus d'alignement de dimensions. Dans cette section, nous effectuons des évaluations expérimentales sur la faisabilité d'aligner des dimensions via différentes mesures de similarité de chaînes de caractères.

6.1. Protocole

6.1.1. Objectifs des expérimentations

Dans la littérature scientifique, de nombreuses mesures de similarité ont été proposées. Cependant, jusqu'à présent, quand et comment utiliser une mesure de similarité reste une question ouverte. Face à ce problème, nous étudions dans cette section l'efficacité et l'efficience des mesures de similarité en fonction de différents paramétrages. L'objectif de nos expérimentations consiste à répondre aux questions suivantes concernant le choix de mesures adéquates dans différents contextes :

– Existe-il une mesure qui a de meilleures performances que les autres indépendamment des stratégies d'alignement de dimensions ?

– Quels sont les impacts des différentes stratégies d'alignement et des seuils sur l'efficacité et l'efficience d'alignement de dimensions ?

– Quelles sont les meilleures mesures en termes de qualité/rapidité, c'est-à-dire les mesures ayant le rapport F-mesure/temps d'exécution le plus élevé ?

6.1.2. Méthode

Durant nos expérimentations, nous comparons 16 mesures de similarité de chaînes de caractères applicables dans le contexte de l'alignement de dimensions. Une librairie Java [5] développée par l'Université UK Sheffield fournit une implantation normalisée des mesures : le résultat final de chaque mesure est compris entre 0 et 1. Pour chaque couple de chaînes de caractères à comparer, des méthodes de prétraitement des chaînes de caractères sont appliquées, telles que transformer toutes les caractères en minuscule, enlever les ponctuations, etc. Pour chaque mesure, nous calculons sa F-mesure en faisant varier le seuil de similarité de 0 à 1. Rappelons qu'un seuil est une valeur définie par l'utilisateur qui sert à éliminer les alignements ayant une similarité inférieure à ce seuil. Nous notons aussi le temps d'exécution nécessaire pour produire tous les résultats finaux.

5. https://sourceforge.net/projects/simmetrics/

Nous distinguons deux groupes d'alignement de dimensions. Le premier groupe se centre sur le schéma des dimensions et repose sur deux stratégies : la stratégie *Concaténée* consiste à construire une longue chaîne de caractères en englobant toutes les instances des attributs d'une dimension ; la stratégie *Séparée* nécessite de spécifier explicitement les correspondances entre les attributs d'un couple de dimensions avant de calculer la similarité. Le deuxième groupe se centre sur les instances des dimensions : la stratégie *Brute* consiste à utiliser les labels d'attributs en tant que tels (instances d'attributs directement extraites d'une sources), tandis que la stratégie *Nettoyée* consiste à enlever le descripteur de type de données au sein d'une instance d'attribut provenant des sources de DOL (comme par exemple `xsd:string`). Les combinaisons des groupes de stratégies d'alignement avec exemples sont disponibles dans la figure 12.

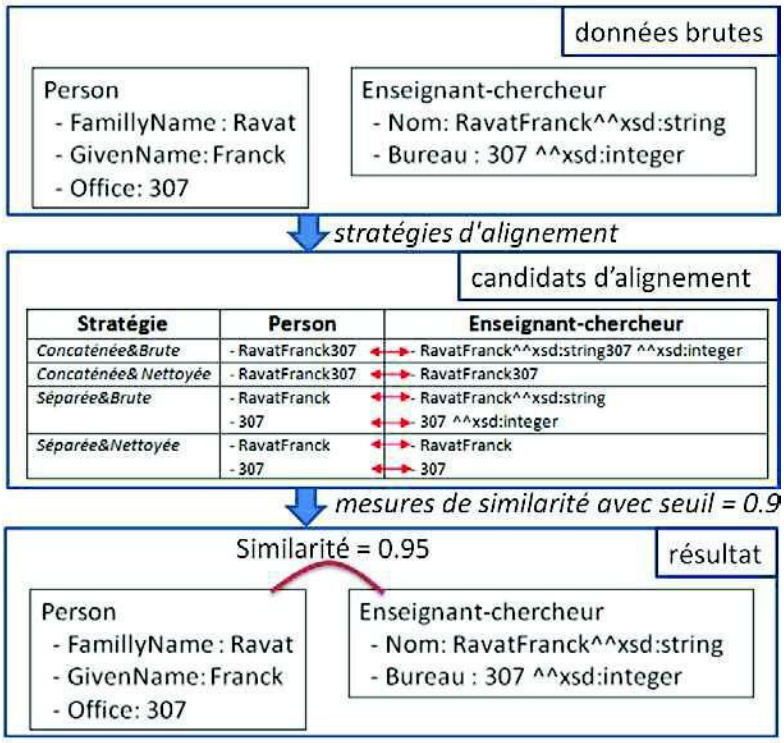


Figure 12. Exemple de l'alignement

6.1.3. Jeux de données

Afin de montrer la faisabilité de construire un *Cube Unifié*, nous effectuons les expérimentations avec des données réelles publiées par le *Département des*

Communautés et du Gouvernement Local (***CLG***)[6] d'Angleterre. Cet organisme met à disposition du grand public des données multidimensionnelles relatives aux logements sociaux, à la taxation des gouvernements locaux, etc. La plupart des données du CLG peuvent être analysées selon deux axes, à savoir l'axe temporel et l'axe géographique. L'axe géographique est composé d'un seul niveau *DISTRICT*, qui peut être complété par l'ajout d'informations complémentaires publiées par le *Bureau de la Statistique Nationale* (*ONS*)[7].

Les sources CLG et ONS contiennent 243 instances de *DISTRICT*. L'objet de nos expérimentations consiste à associer chaque instance de *DISTRICT* de la source CLG à une instance équivalente de la source ONS. Ainsi nous obtenons près de 60 000 (243*243) mappings possibles entre les deux sources. Comme indiqué dans la figure 13, tous les *DISTRICTS* n'ont pas la même représentation.

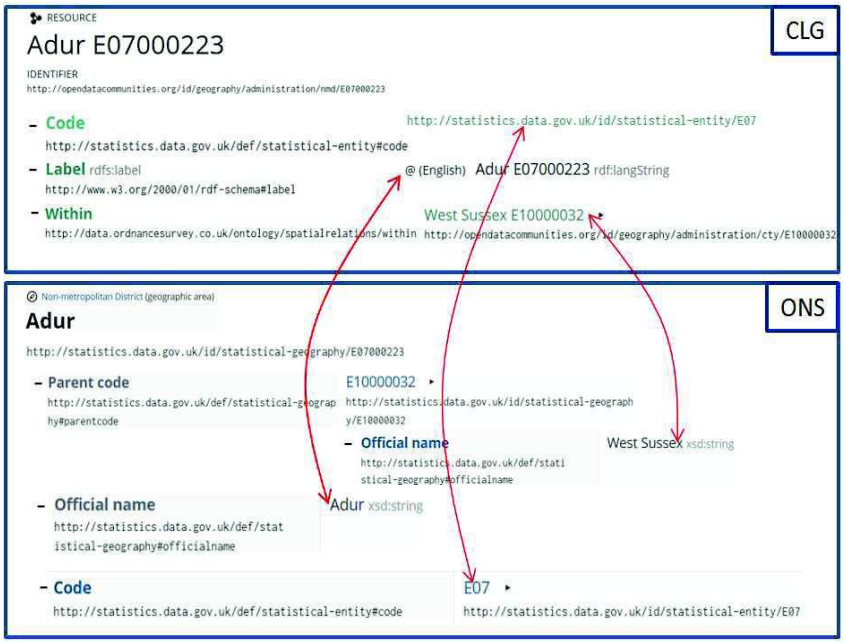


Figure 13. Exemple de DOL en corrélation provenant de différentes sources

– tous les *DISTRICTS* sont décrits à l'aide d'un *code* de classification standardisé (par exemple, `http://statistics.data.gov.uk/id/statistical-entity/E07`) ;

– dans la source CLG, le nom d'un *DISTRICT* est composé d'une chaîne de caractères, d'un numéro de notation et d'un descripteur de type de données étiqueté

6. http://opendatacommunities.org/
7. http://statistics.data.gov.uk/

par la langue utilisée (par exemple, `@(English) Adur E07000223 rdf:langString`). Le nom d'un *DISTRICT* provenant de la source ONS dispose d'une chaîne de caractères et d'un descripteur de type de données (comme par exemple, `Adur xsd:string`) ;

– le niveau parent d'un *DISTRICT* dans la source CLG est décrit par une chaîne de caractères associée à un numéro (par exemple, `West Sussex E10000032`), tandis que le même niveau parent dans la source ONS est représenté par une chaîne de caractères et un descripteur de type de données (tel que `West Sussex xsd:string`).

Dans la source CLG, les *DISTRICTS* sont associés par défaut à ceux de la source ONS à l'aide des liens *owl:sameAs*. Sans tenir en compte de ces liens lors des calculs de similarité, nous en bénéficions en tant que points de comparaison pour évaluer le résultat obtenu par chaque mesure de similarité.

6.1.4. Environnement d'exécution

Respectant la nature distribuée des données dans le contexte du *Cube Unifié*, nous extrayons les données directement depuis les sources via des interfaces d'interrogation[8], sans les matérialiser localement. Une station de travail (Microsoft Windows 7, Intel (R) i7-4510U 2GHz CPU, 8 Go de RAM, disque SSD de 500Go) est utilisée pour calculer la similarité entre les données de deux sources.

6.2. Résultats

6.2.1. L'efficacité : F-mesure

Le premier test consiste à comparer la F-mesure obtenue par chaque mesure de similarité selon différentes stratégies d'alignement et différents seuils.

Nous nous concentrons d'abord sur l'influence des stratégies d'alignement sur la F-Mesure. Avant les expérimentations, notre intuition était la suivante : la F-mesure devait être la meilleure avec la stratégie *Séparée&Nettoyée*, c'est-à-dire la stratégie qui compare séparément les instances sans les descripteurs de type de données. Après exécution, afin de faciliter la lecture, nous choisissons les cinq mesures de similarité ayant la F-mesure la plus élevée pour chaque stratégie d'alignement. Les mesures de similarité dans la figure 14 sont classées en fonction de la F-mesure obtenue. Nous pouvons constater que :

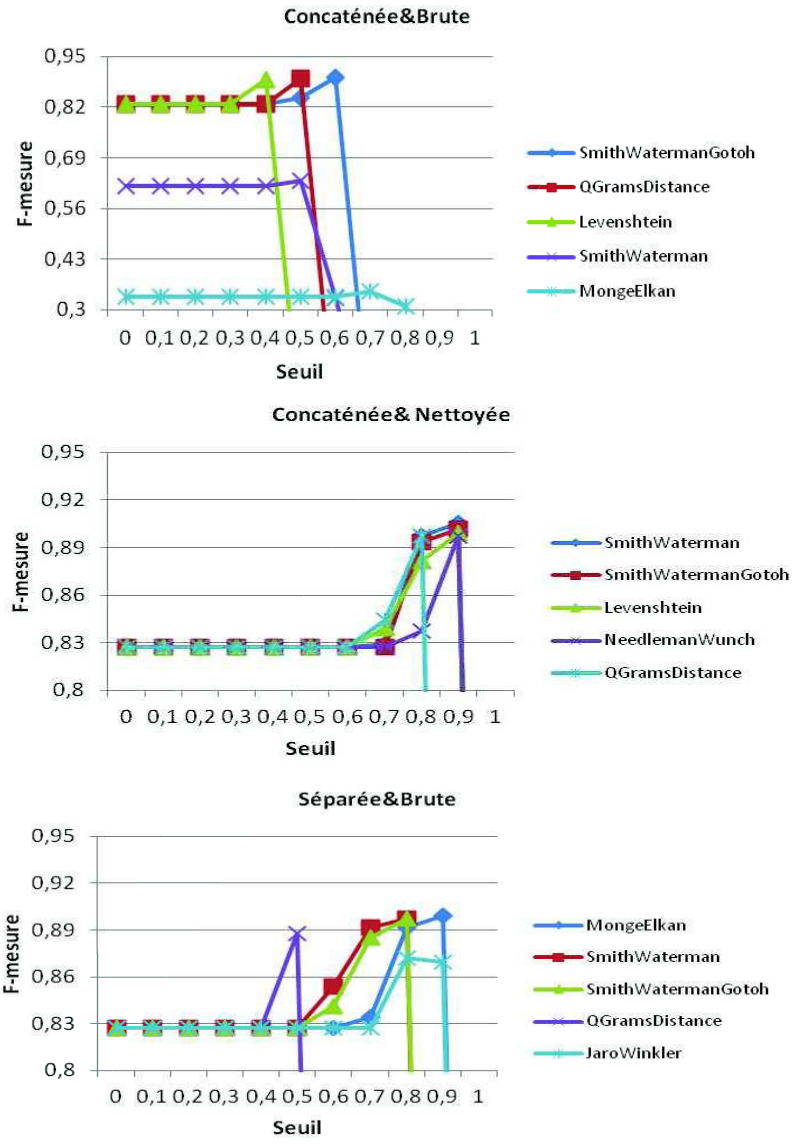
– selon les quatre stratégies d'alignement, la mesure de similarité produisant la meilleure F-mesure varie ;

– parmi les cinq meilleures mesures de similarité de chaque stratégie d'alignement, il y en a trois en commun, à savoir *Q Grams Distance*, *Smith Waterman* et *Smith Waterman Gotoh* (une version améliorée de la mesure *Smith Waterman* proposée par Osamu Gotoh) ;

8. http://opendatacommunities.org/sparql *et* http://statistics.data.gov.uk/sparql

– la F-mesure la plus élevée 0.905 est obtenue par la stratégie *Concaténée&Nettoyée* avec la mesure *Smith Waterman* ;

– les stratégies d'alignement qui ont produit la F-mesure la plus basse (0,897) sont *Concaténée&Brute* et *Séparée&Brute*, qui n'éliminent pas les bruits introduits par le descripteur de type de données.



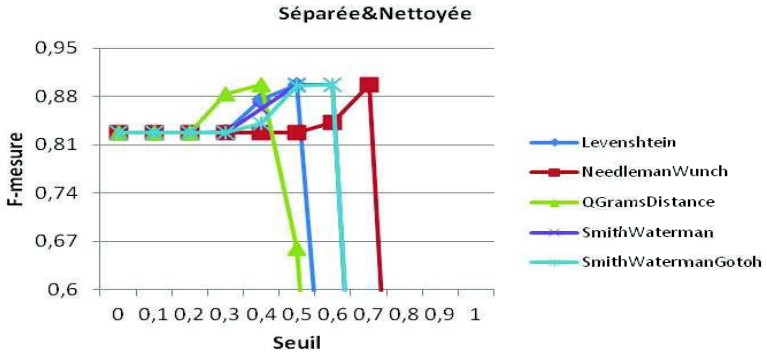


Figure 14. Influences du seuil et de la stratégie d'alignement sur la F-mesure

Nous focalisons maintenant notre étude sur les impacts des seuils sur la F-mesure. Globalement, la F-mesure de chaque mesure de similarité devient plus élevée quand le seuil augmente mais seulement jusqu'à un certain seuil dit *optimal* : un seuil qui permet d'obtenir la meilleure moyenne harmonique de la précision et du rappel. Au-delà de ce seuil *optimal*, la F-mesure chute rapidement car peu de candidats d'alignement ont une similarité supérieure au seuil. Il est important de noter que toutes les mesures de similarité n'ont pas le même seuil *optimal*.

La meilleure F-mesure obtenue pour une même mesure de similarité varie selon les stratégies d'alignement. Premièrement, le choix de concaténer ou pas les instances des attributs dans une dimension a peu d'impact sur les seuils optimaux**.** Deuxièmement, éliminer les descripteurs de type de données permet d'augmenter les seuils optimaux de 30 à 50 %. Notamment, dans le groupe stratégies *Bruitées*, la plupart des mesures de similarité produisent la meilleure F-mesure avec les seuils compris entre 0,4 à 0,7, tandis que dans le group stratégies *Nettoyées*, la majorité des mesures de similarité obtiennent la meilleure F-mesure pour l'intervalle [0,7 ; 0,9].

Nous étudions maintenant l'influence des stratégies d'alignement sur chaque mesure de similarité. Nous choisissons la meilleure F-mesure produite par chaque mesure de similarité selon les stratégies d'alignement. A première vue, toutes les stratégies d'alignement n'ont pas le même impact sur l'efficacité des mesures de similarité (cf. figure 15) :

– les mesures de similarité basées 1) sur vecteur (*Dice Similarity*, *Euclidean Distance*, *Jaccard Similarity*, *Matching Coefficient* et *Overlap Coefficient*), 2) sur termes (*Block Distance* et *Cosine Similarity*) et, 3) sur prononciation phonétique (*Soundex*) produisissent de meilleurs résultats quand le groupe de stratégies *séparées* est appliqué ;

– les mesures de similarité *Jaro* et son extension *Jaro Winkler* obtiennent une F-mesure plus élevée quand les techniques de nettoyage sont appliquées au groupe de

stratégie concaténée. Concernant le groupe de stratégie *Séparée*, le nettoyage a peu d'influence sur l' efficacité de ces mesures ;

– les mesures basées sur sous-séquences de chaînes de caractères (*Q Grams Distance*) sont insensibles aux stratégies d'alignement. Idem pour la majorité des variantes de la mesure de distance d'édition (*Levenshtein*, *Smith Waterman* et *Smith Waterman Gotoh*). Elles font partie des mesures de similarité les plus efficaces avec la moyenne de F-mesure autours de 0,88 ;

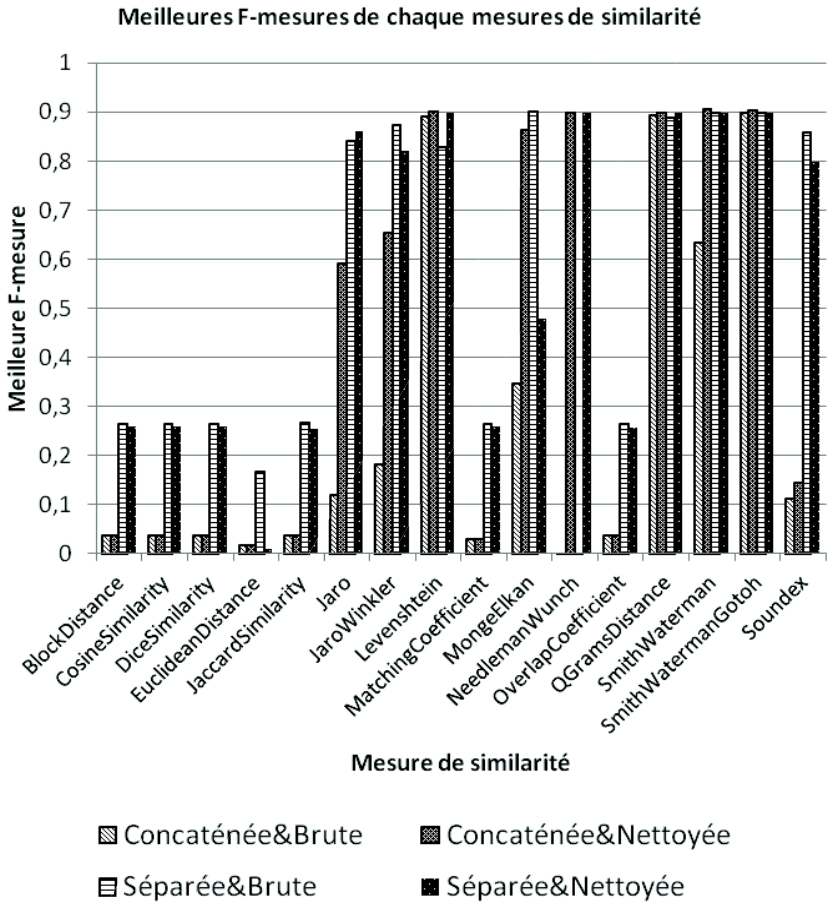


Figure 15. Meilleures F-mesures de chaque mesure de similarité selon les stratégies d'alignement

– deux mesures de similarité basées sur la distance d'édition, à savoir *Monge Elkan* et *Needleman Wunch*, sont plus sensibles aux stratégies d'alignement que les autres. La mesure *Monge Elkan* accorde un coût relativement bas à une séquence

d'insertions ou de suppressions, tandis que la mesure *Needleman Wunch* pénalise plus des insertions et des suppressions de caractères en affectant un coût plus élevé. Par conséquent, la mesure *Monge Elkan* a tendance à « ignorer » les différences entre les chaînes de caractères plus courtes, ce qui rend la F-mesure plus basse en cas de la stratégie *Séparée&Nettoyée*. En revanche, la mesure *Needleman Wunch* est plus appropriée pour identifier des correspondances parfaites entre les chaînes de caractères (en cas du groupe de stratégies *Nettoyées*).

6.2.2. L'efficience : F-Mesure et Temps d'exécution

Dans cette deuxième série de tests, notre objectif est de trouver les mesures de similarité les plus efficientes (produisant la meilleure F-mesure en moins de temps) selon les stratégies d'alignement. Nous utilisons le seuil *optimal* pour chaque mesure de similarité afin d'obtenir la F-mesure la plus élevée. Le tableau 3 montre les cinq mesures de similarité les plus efficaces selon stratégies d'alignement.

Tableau 3. Les mesures de similarité les plus efficaces selon les stratégies d'alignement

Mesures de similarité	***Concaténée &Brute***	***Concaténée &Nettoyée***	***Séparée &Brute***	***Séparée &Nettoyée***
Jaro Winkler			×	
Levenshtein	×	×		×
Monge Elkan	×		×	
Needleman Wunch		×		×
Q Grams Distance	×	×	×	×
Smith Waterman	×	×	×	×
Smith Waterman Gotoh	×	×	×	×

Pour chaque mesure de similarité dans le tableau 3, nous mesurons le temps (en seconds) nécessaires pour son exécution selon différentes stratégies d'alignement. Le résultat est le suivant.

Comme nous pouvons le voir dans la figure 16, le gain en temps d'exécution de certaines mesures telles que *Smith Waterman Gotoh* est important (de 85 % à 500 %) selon les stratégies d'alignement. La stratégie *Séparée&Nettoyée* (qui compare séparément les chaînes de caractères correspondantes en éliminant les descripteurs de type de données) donne le temps d'exécution le plus petit. Cependant, cette stratégie nécessite des connaissances du domaine afin de repérer les attributs correspondants et de spécifier les instances utiles pour le calcul de similarité. Par contre, la stratégie *Concaténée&Brute* (qui concatène toutes les instances d'attributs d'une dimension sans les nettoyer) donne le temps d'exécution le plus élevé. Même

si le temps d'exécution de certaines mesures devient long, cette stratégie permet de réduire l'intervention des utilisateurs lors du calcul de similarité, tout en garantissant la qualité des résultats obtenus (cf. Introduction).

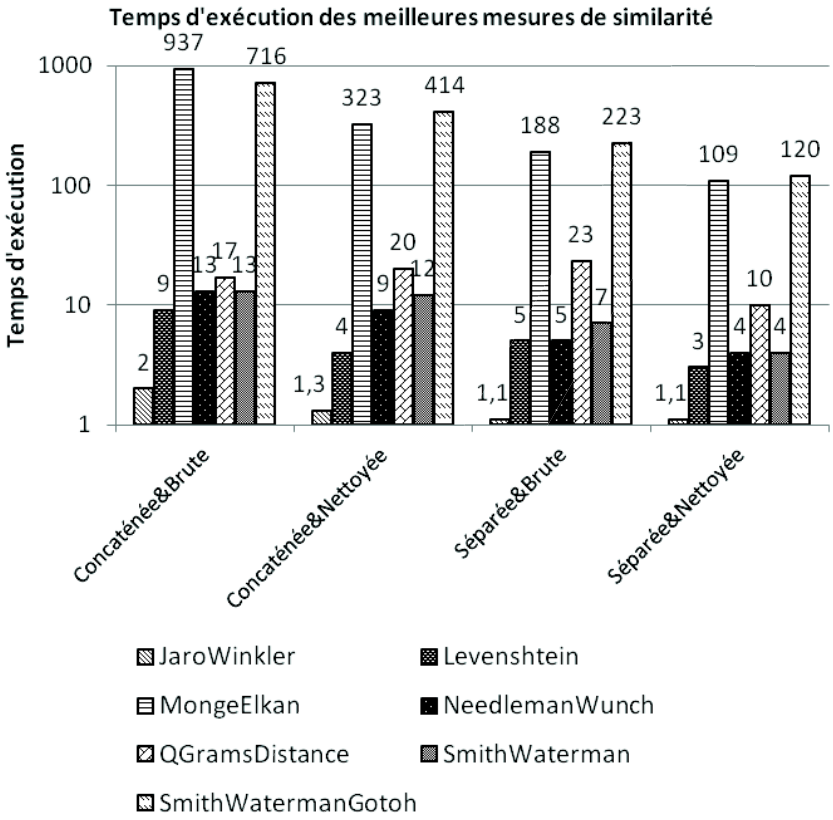


Figure 16. Temps d'exécution des meilleures mesures de similarité selon les stratégies d'alignement

Nous focalisons maintenant notre étude sur l'efficience des mesures de similarité. Même si *Smith Waterman Gotoh* et *Monge Elkan* font partie des mesures de similarité les plus efficaces, leur rapport F-mesure/temps d'exécution est bas à cause d'une exécution très chronophage. Elles sont donc appropriées aux calculs de similarité des chaînes de caractères courtes. *JaroWinkler* permet de produire rapidement un résultat (en moins de 2 secondes). Cependant, comme la measure favorise les chaînes de caractères partageant un préfixe commun, sa gamme d'application reste assez étroite et nécessite d'appliquer la stratégie *Séparée* lors de l'alignement de dimensions. Enfin, *Levenshtein, Needleman Wunch, Q Grams Distance* et *Smith Waterman* sont des mesures de similarité ayant les plus grands

rapports de F-mesure/temps d'exécution : elles permettent d'obtenir les meilleures F-mesures avec des temps polynomiaux.

6.3. Conclusions des évaluations expérimentales

Suite à nos expérimentations, les conclusions sont les suivantes. Premièrement, toutes les mesures de similarité ne sont pas adaptées à l'alignement de dimensions. Certaines mesures (*Overlap Coefficient*, *Soundex,* etc.) sont conçues pour des besoins spécifiques (cf. Christian, 1998 ; Cohen, Ravikumar, et Fienberg, 2003 ; Ding *et al.*, 2008) et il n'est pas judicieux de les utiliser dans un contexte général comme nous souhaitons le faire.

Deuxièmement, les mesures de similarité produisant les meilleurs résultats d'alignement de dimensions sont *Jaro Winkler*, *Levenshtein*, *Monge Elkan*, *Needleman Wunch*, *Q Grams Distance*, *Smith Waterman* et *Smith Waterman Gotoh*. Il faut toutefois combiner ces mesures avec des stratégies d'alignement adéquates afin de maximiser leur efficacité. Pour un utilisateur non-expert qui peut appliquer uniquement la stratégie *Concaténée&Brute*, ses choix se limitent aux mesures *Levenshtein*, *Q Grams Distance*, *Smith Waterman* et *Smith Waterman Gotoh*. Par contre, pour un utilisateur expert qui souhaite utiliser les différentes stratégies d'alignement, la liste ci-dessus peut être complétée par *Jaro Winkler* (avec *Séparée&Brute*), *Monge Elkan* (avec *Séparée&Brute*) et *Needleman Wunch* (avec *Concaténée& Nettoyée* et *Séparée&Nettoyée*).

Enfin, si nous exigeons un temps d'exécution court tout en garantissant l'efficacité des mesures, un utilisateur non expert peut choisir *Levenshtein*, *Q Grams Distance*, *Smith Waterman*. Un utilisateur expert peut également utiliser la mesure *Needleman Wunch* en complément de la liste précédente.

7. Analyse Décisionnelle avec un *Cube Unifié*

Afin de valider la faisabilité de notre proposition, nous explicitons dans cette section le principe des analyses décisionnelles sur un *Cube Unifié*.

7.1. Processus d'analyse

Dans nos travaux précédents (Ravat et Song, 2016), nous présentons en détail un framework permettant aux décideurs d'effectuer des analyses avec un *Cube Unifié*. En raison du peu d'espace disponible, nous donnons ici une vue d'ensemble sur le processus d'analyse dans le framework.

Comme indiqué dans la figure 17, durant une analyse, un décideur exprime un besoin via le schéma d'un *Cube Unifié*. Ce besoin d'analyse est automatiquement traduit en plusieurs requêtes ; chaque requête est associée et exécutée dans une seule source, chaque source produit un résultat partiel ; enfin, le système se réfère aux

relations corrélatives pour aligner les instances de dimension. Ainsi, un résultat d'analyse unique est produit en unifiant tous les résultats partiels produits indépendamment par les sources. Dans (Ravat et Song, 2016), vous trouverez des détails complémentaires sur le framework permettant aux décideurs d'effectuer des analyses avec un *Cube Unifié*.

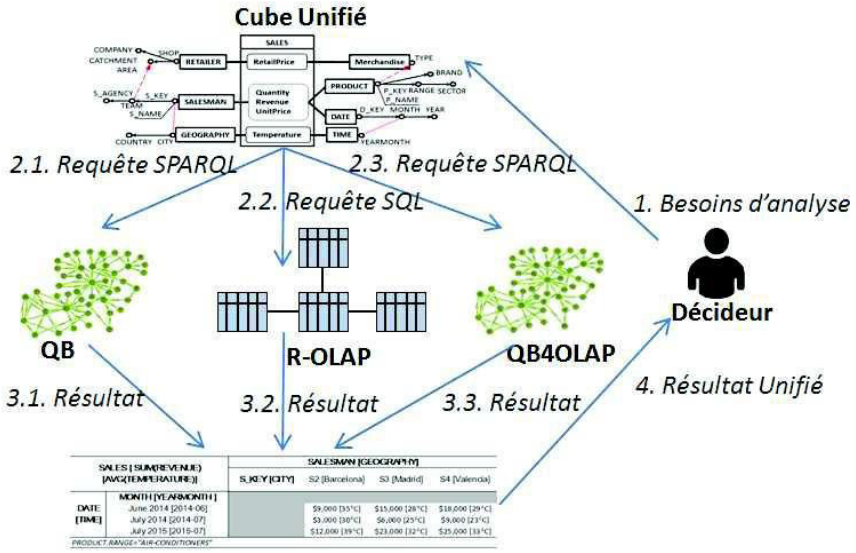


Figure 17. Processus d'analyse des données provenant de multiple sources

Dans notre cas d'étude, les sources sont implantées de manière suivante : le SGBD Oracle 11g héberge les données entreposées, tandis que le SGBD *triple-store* fourni par Apache Jena API (cf. Carroll *et al.*, 2004) est utilisé pour gérer des DOL aux formats QB et QB4OLAP. Le schéma du *Cube Unifié* est implanté dans un méta-modèle *via* un programme. Une table de correspondance est utilisée pour stocker les *relations corrélatives* manuellement construites entre les dimensions.

7.2. Besoin d'analyse et son résultat

Supposons qu'un décideur souhaite connaître « *la quantité des radiateurs vendus selon le pays dans lequel les vendeurs travaillent* ». Pour répondre à ce besoin, des données de l'ED R-OLAP doivent être agrégées selon un axe d'analyse provenant de la source QB. Cette analyse multi-sources est possible grâce à la *relation corrélative* entre la dimension *SALESMAN* et la dimension *GEOGRAPHY* (voir figure 10b). Deux requêtes sont générées durant l'analyse. Afin d'être indépendant de tous langages d'interrogation, les requêtes sont présentées sous une forme algébrique. De plus, bien que l'algèbre SPARQL ne soit pas encore un standard du

W3C, elle est déjà intégrée dans plusieurs SGBD de DOL et toutes les expressions algébriques SPARQL sont générées automatiquement par Apache Jena API.

Nous pouvons voir dans la figure 18 que la requête SQL renvoie la quantité des radiateurs vendus selon les vendeurs, tandis que la requête SPARQL trouve la liste des pays dans la source QB.

```
S_Key ℱsum(quantity)((Sales JOIN P_Key=P_Key
 SELECTRange='Heaters'(Product)))

(PROJECT (?country)
 (BGP
   (TRIPLE ?geo a qb:DimensionProperty)
   (TRIPLE ?geo rdfs:label "Geography"@en)
   (TRIPLE ?geo qb:codeList ?lstP)
   (TRIPLE ?lstP skos:hasTopConcept ?pCountry)
   (TRIPLE ?pCountry a skos:Concept)
   (TRIPLE ?pCountry rdfs:label "Country"@en)
   (TRIPLE ?insCountry a ?pCountry)
   (TRIPLE ?insCountry rdfs:label ?country)))
```

Figure 18. Requêtes générées pour la première analyse

En nous référant aux *relations corrélatives*, les instances en corrélation de la dimension *SALESMAN* et de la dimension *GEOGRAPHY* sont reliées ensemble. Cela permet d'offrir des perspectives d'analyse supplémentaires aux décideurs : comme indiqué dans la figure 19, les vendeurs qui ont vendu moins de radiateurs se situent dans des pays chauds où la demande reste très faible. Sans le *Cube Unifié*, il serait difficile d'obtenir de multiples perspectives si l'analyse était effectuée séparément avec des sources indépendantes.

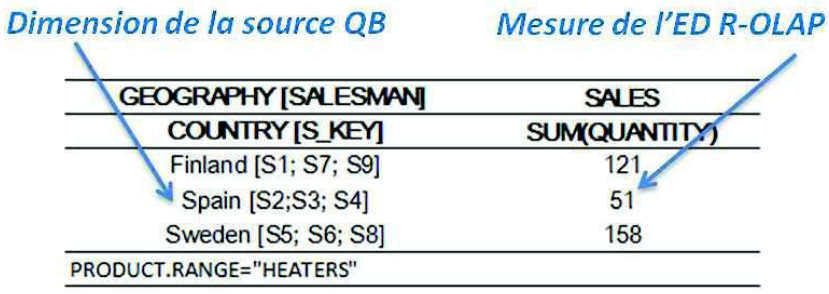


Figure 19. Résultat unifié de l'analyse

8. Conclusion

Notre objectif est d'offrir un environnement contenant toutes les données pertinentes pour les prises de décision. À cette fin, nous représentons dans un schéma unifié, appelé *Cube Unifié*, des données entreposées et des DOL multidimensionnelles. Un *Cube Unifié* repose sur des concepts orientés utilisateur et contient des notations graphiques afin de faciliter les analyses décisionnelles.

Pour construire un *Cube Unifié*, nous décrivons un processus en deux étapes qui unifie les données sources. Dans un premier temps, les schémas publiés avec des langages de modélisation spécifiques sont transformés en une représentation conceptuelle appelée *cube d'exportation*. Définis à l'aide d'un langage de modélisation générique, les *cubes d'exportation* permettent de faciliter la combinaison de données sources dans un schéma unifié. La deuxième étape consiste à associer des *cubes d'exportation* en alignant les dimensions en corrélation. Le *Cube Unifié* comprend un ensemble de *relations corrélatives* représentant les correspondances entre les dimensions. Un opérateur de liaison est proposé pour établir ces *relations corrélatives* entre deux dimensions .

Pour valider la faisabilité de l'approche, des évaluations expérimentales ont été effectuées avec des jeux de données réelles. Parmi l'ensemble des mesures de similarité, nous avons identifié les quatre mesures de similarité les plus efficaces et les plus efficientes pour identifier des *relations corrélatives*, à savoir *Levenshtein, Needleman Wunch, Q Grams Distance* et *Smith Waterman*. Pour vérifier la véracité de nos propositions, nous avons proposé un framework d'analyse décisionnelle sur des sources multiples. Ce framework d'analyse a permis de montrer qu'un *Cube Unifié* peut fournir de multiples perspectives complémentaires à un décideur durant une analyse décisionnelle.

Un *Cube Unifié* unifie des données internes et/ou externes dont le décideur ne connaît pas a priori la sémantique. Des mesures de similarité jouent un rôle important pour identifier les *relations corrélatives* entre ces différentes données. Ainsi, un de nos travaux en cours consiste à proposer des lignes directrices sur les choix des mesures de similarité selon divers critères, tels que la longueur des chaînes de caractères, le type de données, etc. Une autre perspective est d'étudier l'efficience d'unifier aussi bien des données entreposées que des DOL ayant des volumétries différentes. Pour ce faire, nous devons effectuer d'autres expérimentations avec un banc d'essais adéquat. Un objectif à long terme est d'intégrer d'autres types de relations entre dimensions dans un *Cube Unifié*. Par exemple, une relation d'agrégation serait utile pour représenter les liens *père-fils* entre deux dimensions. Des relations holistiques devront aussi être proposées afin de relier n dimensions ($n \geq 2$).

Bibliographie

Abelló A., Darmont J., Etcheverry L., Golfarelli M., Mazón J.-N., Naumann F. *et al.* (2013). Fusion Cubes: Towards Self-Service Business Intelligence. *International Journal of Data Warehousing and Mining*, vol. 9, n° 2, p. 66-88.

Abelló A., Romero O., Pedersen T. B., Berlanga R., Nebot V., Aramburu M. J. *et al.* (2015, février). Using SemanticWeb Technologies for Exploratory OLAP: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, n° 2, p. 571-588.

Bauer F., Kaltenböck M. (2012). *Linked open data: the essentials: a quick start guide for decision makers*. Wien. (OCLC: 802548205).

Bellatreche L., Khouri S., Berkani N. (2013). Semantic Data Warehouse Design: From ETL to Deployment Ã la Carte. In *Database Systems for Advanced Applications*, vol. 7826, Berlin, Heidelberg, Springer Berlin Heidelberg, p.64-83.

Bernstein P. A., Madhavan J., Rahm E. (2011). Generic schema matching, ten years later. *Proceedings of the VLDB Endowment*, vol. 4, n° 11, p. 695-701.

Cabanac G., Max C., Ravat F., Teste O. (2009). Decisional Annotations: Integrating and Preserving Decision-Makers' Expertise in Multidimensional Systems. In *Complex Data Warehousing and Knowledge Discovery for Advanced Retrieval Development: Innovative Methods and Applications*, p. 65-81. IGI Global.

Carroll J. J., Dickinson I., Dollin C., Reynolds D., Seaborne A., Wilkinson K. (2004). Jena: implementing the semantic web recommendations. In *Proceedings of the 13th international World Wide Web conference*, p. 74-83. New York, NY, USA, ACM Press.

Chaudhuri S., Dayal U. (1997, mars). An overview of data warehousing and OLAP technology. *ACM SIGMOD Record*, vol. 26, n° 1, p. 65-74.

Christian P. (1998). Soundex-can it be improved? *Computers in Genealogy*, vol. 6, p. 215-221.

Cohen W., Ravikumar P., Fienberg S. (2003). A comparison of string metrics for matching names and records. In *Kdd workshop on data cleaning and object consolidation*, vol. 3, p.73-78.

Deb Nath R. P., Hose K., Pedersen T. B. (2015). Towards a Programmable Semantic Extract-Transform-Load Framework for Semantic Data Warehouses. In *Proceedings of the ACM Eighteenth International Workshop on Data Warehousing and OLAP*, p. 15-24. ACM Press.

Ding H., Trajcevski G., Scheuermann P., Wang X., Keogh E. (2008, août). Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, vol. 1, n° 2, p. 1542-1552.

Doan A., Halevy A. Y. (2005). Semantic integration research in the database community: A brief survey. *AI magazine*, vol. 26, n° 1, p. 83-94.

Etcheverry L., Vaisman A., Zimányi E. (2014). Modeling and Querying Data Warehouses on the Semantic Web Using QB4olap. In *Data Warehousing and Knowledge Discovery*, vol. 8646, p. 45-56. Cham, Springer International Publishing.

Etcheverry L., Vaisman A. A. (2012). Enhancing OLAP Analysis with Web Cubes. In *The Semantic Web: Research and Applications*, vol. 7295, Springer Berlin Heidelberg, p. 469-483.

Euzenat J. (2013). *Ontology matching* (2nd edition éd.). New York, Springer.

Fleischhacker D., Paulheim H., Bryl V., Völker J., Bizer C. (2014). Detecting Errors in Numerical Linked Data Using Cross-Checked Outlier Detection. In *The Semantic Web ISWC 2014*, vol. 8796, p. 357-372. Cham, Springer International Publishing.

Ghozzi F., Ravat F., Teste O., Zurfluh G. (2005). Méthode de conception d'une base multidimensionnelle contrainte. In *1ère journée francophone sur les Entrepôts de Données et l'Analyse en ligne*, EDA 2005, p. 51-70. Lyon, France.

Kämpgen B., Harth A. (2011). Transforming statistical linked data for use in OLAP systems. In *Proceedings of the 7th international conference on Semantic systems*, p. 33-40. Graz, Austria, ACM Press.

Laborie S., Ravat F., Song J., Teste O. (2015, mai). Combining Business Intelligence with Semantic Web: Overview and Challenges. In *INFormatique des Organisations et Systèmes d'Information et de Décision (INFORSID'15)*. Biarritz, France.

Matei A., Chao K.-M., Godwin N. (2015). OLAP for Multidimensional Semantic Web Databases. In *Enabling Real-Time Business Intelligence*, vol. 206, p. 81-96. Springer Berlin Heidelberg.

Nebot V., Berlanga R., Pérez J. M., Aramburu M. J., Pedersen T. B. (2009). Multidimensional Integrated Ontologies: A Framework for Designing Semantic Data Warehouses. In *Journal on Data Semantics XIII*, vol. 5530, p. 1-36. Springer Berlin Heidelberg.

Rahm E. (2011). Towards Large-Scale Schema and Ontology Matching. In *Schema Matching and Mapping*, p. 3-27. Springer Berlin Heidelberg.

Rahm E., Bernstein P. A. (2001, décembre). A survey of approaches to automatic schema matching. *The VLDB Journal*, vol. 10, n° 4, p. 334-350.

Ravat F., Song J. (2016). Enabling OLAP Analyses on the Web of Data. In *IEEE The Eleventh International Conference on Digital Information Management*, p. 215 – 224. Porto, Portugal, IEEE.

Romero O., Abelló A. (2007). Automating multidimensional design from ontologies. In *international workshop on Data warehousing and OLAP*, p. 1-8. ACM Press.

Saad R., Teste O., Trojahn C. (2013). OLAP Manipulations on RDF Data following a Constellation Model. *In Proceedings of International Workshop on Semantic Statistics (SemStats2013) collocated with International Semantic Web Conference (ISWC 2013)*. Sydney.

Skoutas D., Simitsis A. (2007). Ontology-Based Conceptual Design of ETL Processes for Both Structured and Semi-Structured Data. *International Journal on Semantic Web and Information Systems*, vol. 3, n° 4, p. 1-24.

Thenmozhi M., Vivekanandan K. (2014, juin). An Ontological Approach to Handle Multidimensional Schema Evolution for Data Warehouse. *International Journal of Database Management Systems*, vol. 6, n° 3, p. 33-52.

Zorrilla M. E., Mazón J.-N., Ferrández O., Garrigós I., Daniel F., Trujillo J. (Eds.). (2012). *Business Intelligence Applications and the Web: Models, Systems and Technologies*. IGI Global.