Automatika
Journal for Control, Measurement, Electronics, Computing and Communications

ISSN: 0005-1144 (Print) 1848-3380 (Online) Journal homepage: https://www.tandfonline.com/loi/taut20

The potential of semantic paradigm in warehousing of big data

Marina Ptiček, Boris Vrdoljak & Marko Gulić

To cite this article: Marina Ptiček, Boris Vrdoljak & Marko Gulić (2019) The potential of semantic paradigm in warehousing of big data, Automatika, 60:4, 393-403, DOI: 10.1080/00051144.2019.1630582

To link to this article: https://doi.org/10.1080/00051144.2019.1630582

AUTOMATIKA
2019, VOL. 60, NO. 4, 393–403
https://doi.org/10.1080/00051144.2019.1630582

REGULAR PAPER

OPEN ACCESS Check for updates

The potential of semantic paradigm in warehousing of big data

Marina Ptiček [a], Boris Vrdoljak [a] and Marko Gulić [b]

[a]Faculty of Electrical Engineering and Computing, University of Zagreb, Zagreb, Croatia; [b]Faculty of Maritime Studies Rijeka, University of Rijeka, Rijeka, Croatia

ABSTRACT
Big data have analytical potential that was hard to realize with available technologies. After new storage paradigms intended for big data such as NoSQL databases emerged, traditional systems got pushed out of the focus. The current research is focused on their reconciliation on different levels or paradigm replacement. Similarly, the emergence of NoSQL databases has started to push traditional (relational) data warehouses out of the research and even practical focus. Data warehousing is known for the strict modelling process, capturing the essence of the business processes. For that reason, a mere integration to bridge the *NoSQL gap* is not enough. It is necessary to deal with this issue on a higher abstraction level during the modelling phase. NoSQL databases generally lack clear, unambiguous schema, making the comprehension of their contents difficult and their integration and analysis harder. This motivated involving semantic web technologies to enrich NoSQL database contents by additional meaning and context. This paper reviews the application of semantics in data integration and data warehousing and analyses its potential in integrating NoSQL data and traditional data warehouses with some focus on document stores. Also, it gives a proposal of the future pursuit directions for the *big data warehouse* modelling phases.

ARTICLE HISTORY
Received 4 August 2018
Accepted 6 June 2019

KEYWORDS
Semantic Web technologies; data warehouse; NoSQL database; document store

1. Introduction

For decision makers, having the correct information at the right moment is crucial. Context captured by such information must be analytically relevant to give them additional, *hidden* knowledge needed for quality decision-making. To create such information, data scattered in various storages must often be integrated. Data integration has been dealt with for many years [1], but it is also nowadays' research topic. Data warehousing is an example of integration aimed for analytical purposes. At first, data warehouses (DWHs) were designed for business (retail) analysis but were later applied in other domains as well. Traditionally, DWH's relational implementation starts with the design of its schema, which includes the analysis of involved data sources' schemas.

Big data are becoming more analytically interesting with the availability of commodity hardware and new processing and storage solutions, which changed the perception of data that have not been previously perceived as analytical potential. The chances are that newly generated data will end up in nowadays' data storage trend, NoSQL databases (NoSQL DBs). To unify existing storage and analytical systems with the *new* ones, research focused on NoSQL DB integration with relational databases (RDBs), mostly making a *physical* data integration into one storage. But, for DWH-based *big data* analytics that is not enough because that is only the final step in DWH implementation. NoSQL DBs generally lack clear schema, and this fact has caused discussions on data modelling [2]. It is argued that design should not be skipped, but that it should rather become *evolutionary* and *iterative* [2]. We agree and argue that, to implement a DWH successfully, design process must take place. The lack of NoSQL DB schema imposes integration problems on DWHs, thus finding a way for the integration of *big data* with the existing analytical systems is necessary. To compensate this problem, one research direction is turning to ontologies, sets of concepts and relationships from the Semantic Web. A detailed review of the research on the role of Semantic Web technologies in data warehousing is provided in [3].

Unlike the latter, we focus our research on the use of semantics both in NoSQL data integration and data warehousing. We have reviewed these topics, in a general fashion, in our most recent work [4]. This paper extends our previous work and presents the continuation of our analysis on the mentioned topics. We extend our previous paper with wider literature review and provide its systematization. Accordingly, this paper provides an extended discussion. As the result of our further research, we analyse in detail document stores

CONTACT Marina Ptiček marina.pticek@fer.hr Faculty of Electrical Engineering and Computing, University of Zagreb, Unska 3, 10000 Zagreb, Croatia

as a subset of *big data*-related problems in the data warehousing context and therefore, we extend our analysis to the document store context where the discussion of the related problems is appropriate. One of the main reasons for this decision is the popularity of MongoDB,[1] which is a document-store and we believe that integration problems must be more profoundly tackled in that context. In the discussion we suggest ordering of DWH design phases that is different from all design approaches that featured a NoSQL source so far, and by this methodological suggestion, we pave the way for our further research.

The paper is organized as follows: Section 2 addresses main features of traditional data warehousing; Section 3 reviews current data trends; Section 4 addresses general integration problematics and how it manifests in the context of big data and data warehouses; Section 5 addresses research related to application of semantics in data integration and (big) data warehousing; Section 6 discusses reviewed research and problematics and proposes research topics for future pursuit for *big data warehouse* modelling; and Section 7 concludes this paper.

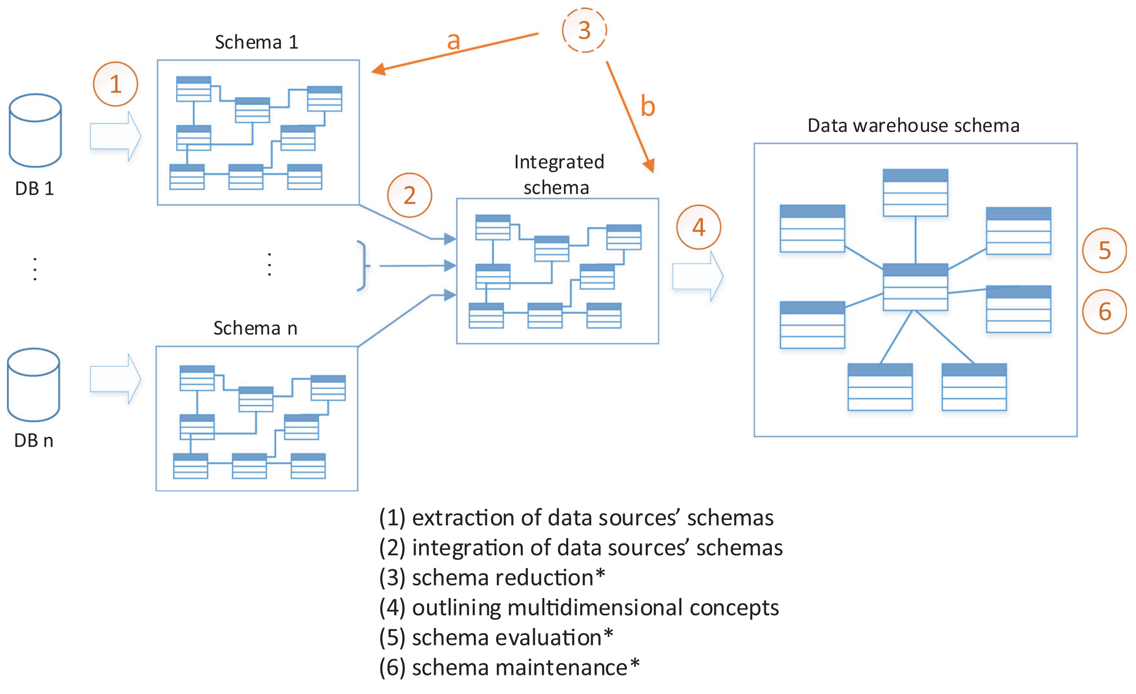
2. *Traditional* data warehouse design

Design is the first step of DWH implementation and precedes its logical modelling, physical implementation, and Extract-Transform-Load (ETL) process. DWH design is roughly divided into phases shown in Figure 1.

DWH design starts with the analysis of the data sources' schemas, which may be consolidated and integrated into a common schema. Next phase is the identification of multidimensional concepts (MDCs) and their composition into a DWH schema, which is done based on either data sources' schemas, the integrated schema, or even data. During these phases, the schema reduction process is present either after all or after certain phases (1 or 2). Schema reduction helps designers refine important concepts for further phases because not all of them are analytically relevant. Once the schema is designed, it should be evaluated to reduce the costs of correction of possible errors spotted during the physical implementation and ETL process. DWH schema maintenance has long been a debatable topic because the change of DWH schema can cause problems if it is not carried out well. However, the value added by bringing a new data source to the DWH is the main reason to permit these changes. The main strategies to handle schema changes are schema evolution (updating DWH schema and data [5]), versioning (maintaining different DWH schema versions [6]), and view maintenance (representing all states of data sources using materialized views).

3. Current data trends

There are four main NoSQL database families – document, column and key-value stores, and graph databases – differing significantly in their features (model, ways of storing data, etc.) thus, there is no unified NoSQL model, nor a standardized query language. Features making them favourable for big data are good scalability and parallel processing on commodity hardware, and – as often pointed out – fast



(1) extraction of data sources' schemas
(2) integration of data sources' schemas
(3) schema reduction*
(4) outlining multidimensional concepts
(5) schema evaluation*
(6) schema maintenance*

*Phases highlighted by an asterisk are either not necessarily applied in order specified by the list or are optional.

Figure 1. Phases of the *traditional* DWH design.

data loading, achieved by *loosened schema* constraints on the data, meaning that the data structure, formatting and contents are not validated against a schema during the loading phase (*schema-on-write*). Schema is rather interpreted later during data usage (*schema-on-read*). However, if schema precision is demanded (just like in DWH modelling or implementing an SQL-like query language), this is a downside. By being the favoured storage for big data, NoSQL DBs inherit their four main *V-features* – *volume*, *velocity*, *variety*, and *veracity* – all problematic for the integration NoSQL with the traditional DWH.

Document stores (also called document-oriented databases) are the most adopted NoSQL paradigm. They store data in forms of documents, i.e. sets of *key-value* pairs, where the *key* part of the data is the metadata ("name tag") of its *value* part. Documents bear great similarity to XML documents for being self-describing and having hierarchy concept – the basic difference is that a JSON document is shorter and easier to read and write. Documents in a document store are grouped in collections. Comparing document and relational storage paradigms conceptually, most often conceptual mapping is that a collection corresponds to a table (relation), a document to a tuple, *keys* to column (attribute) names, and *values* to values stored in *cells*.

4. Integration problems

Accelerated emergence of new paradigms in data processing and storage is a trend that is gradually becoming a problem. System and data integration are also currently active research areas and will remain so in the future, but their complexity will increase with the diversity of emerging technologies and paradigms. Still, from a higher-level perspective, general integration problems remain the same and have been addressed in the following subsections in regard to DWH and NoSQL DB integration.

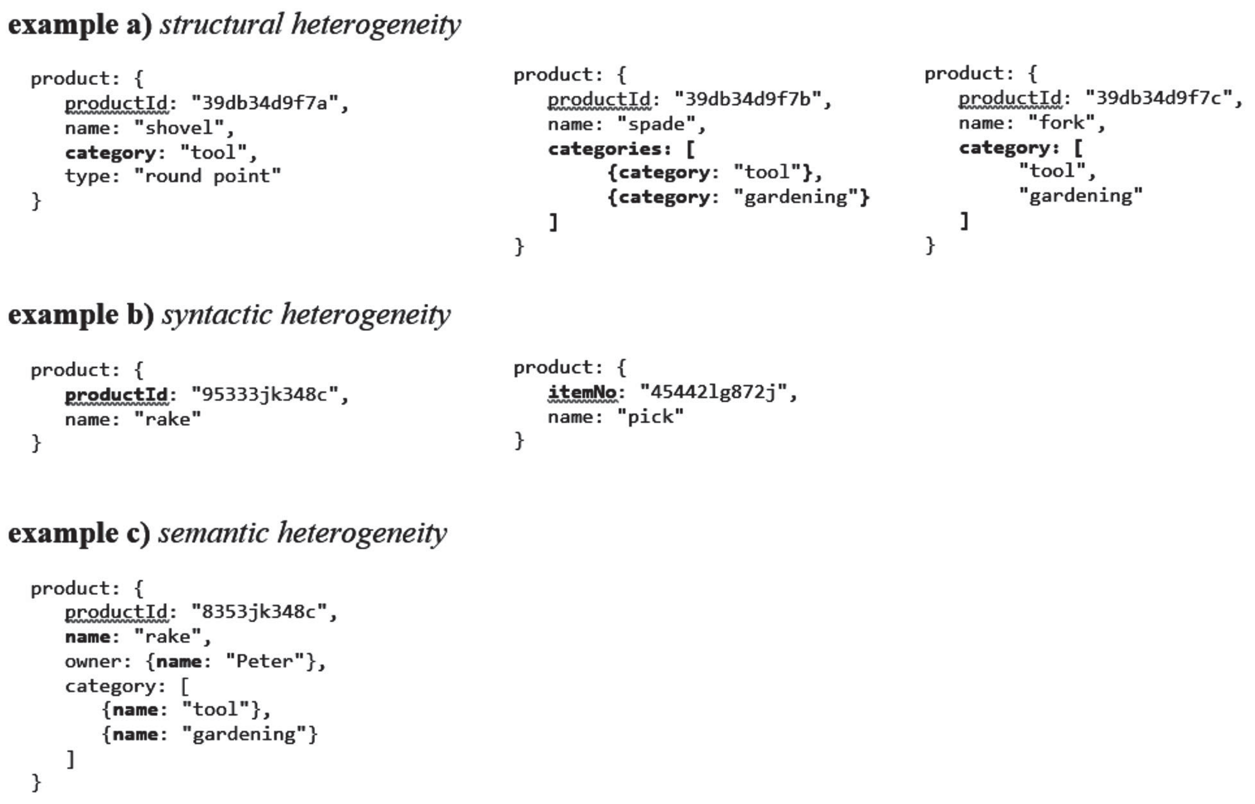
4.1. Heterogeneity

Data integration is often hurdled by heterogeneity, categorized in three types [7]: *structural* (attribute arrangement, nesting, presence or absence), *syntactic* (different naming for the same concept), and *semantic* (same naming of different concepts). Examples in JSON format are shown in Figure 2. Due to *schema-on-read* approach, all three types of heterogeneity can be expected in document stores, as well as other NoSQL DBs, condensed as the big data *variety* feature.

4.2. Integration levels

Integration can be done on three roughly categorized levels: *physical* (pure storage-level/data integration), *middleware/interface* (via common or middleware interface, providing uniform access or using an integration application), and *schema-level* (involving analysis of sources' schemas and schema integration).

Physical integration implies a common data model and storage, thus choosing the target storage type is a



example a) *structural heterogeneity*

```
product: {
   productId: "39db34d9f7a",
   name: "shovel",
   category: "tool",
   type: "round point"
}
```

```
product: {
   productId: "39db34d9f7b",
   name: "spade",
   categories: [
        {category: "tool"},
        {category: "gardening"}
   ]
}
```

```
product: {
   productId: "39db34d9f7c",
   name: "fork",
   category: [
        "tool",
        "gardening"
   ]
}
```

example b) *syntactic heterogeneity*

```
product: {
   productId: "95333jk348c",
   name: "rake"
}
```

```
product: {
   itemNo: "454421g872j",
   name: "pick"
}
```

example c) *semantic heterogeneity*

```
product: {
   productId: "8353jk348c",
   name: "rake",
   owner: {name: "Peter"},
   category: [
      {name: "tool"},
      {name: "gardening"}
   ]
}
```

Figure 2. Heterogeneity in JSON documents.

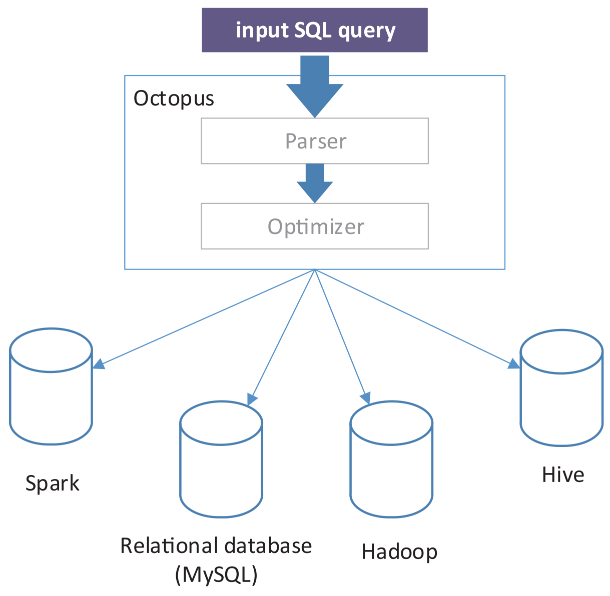


Figure 3. Integration of multiple data storage and processing engines using Octopus [9] middleware.

delicate question. No storage engine is suitable for all data types, especially not all NoSQL data, while traditional storage engines like RDBMSs have proven to deal with relational data best but are unable to handle big data well. Forcing natively relational data into NoSQL DBs is also not a good solution, because it would lead to the loss of orderliness of such data.

Integration using middleware can be applied in various ways, reviewed in our previous work [8]. Middleware can be used to integrate NoSQL DBs, RDBs, and data processing engines such as Spark or Hadoop (Figure 3), with the main idea of hiding complexities of integrated systems and disburdening the users from learning specificities of underlying systems, and minimization of data movement between the processing and storage engines, which is important in case of voluminous data. This is achieved by *query mediation*: user issues a query on a common *virtual schema*, which is then divided into subqueries that are *pushed down* to the storages based on data's residence. Underlying storages return the result sets to the middleware, which unifies them and returns a singular, unified result set to the user.

The DWH design includes conceptual modelling. Through the modelling, DWH's star schemas are designed, meaning that *schema-level integration* is a mandatory process in the design. The modelling process in DWH is of utmost importance because it ensures the inclusion of analytically interesting concepts in the DWH model. For this reason, schema-level integration needs to occur also in the *big data warehouse* (*BigDWH*) design process to sensibly choose which data from NoSQL data source would be warehoused. Due to the aforementioned heterogeneity problems, research needs to be done on adjustment of the traditional modelling process, aiming at the inclusion of NoSQL data.

4.3. Design approach

There are three approaches to DWH design: *data-oriented*, *demand/requirement-driven*, and *hybrid*.

Data-oriented approach guarantees availability of all concepts involved in modelling, but might produce a DWH that is not aligned with the requests.

Requirement-driven approach demands end users' clear vision of the DWH use cases, but it may be discovered in later phases that not all requested concepts are available at the data sources. Also, an opportunity might be missed to discover analytically interesting concepts that were available at the data sources but had not been stated in the requirements.

Hybrid approach is more complex, combining the best features of the previous two because each is flawed in its way. It has shown to be a promising approach for the integration of NoSQL data into a DWH and numerous attempts on its automatization have been made [10–13] because it significantly reduces design efforts. Components of the requirement-driven approach have an important role in reducing large schema reengineered from the NoSQL data source (reducing the dimensionality) and thus focusing the designer's view only on analytically relevant concepts and associated data.

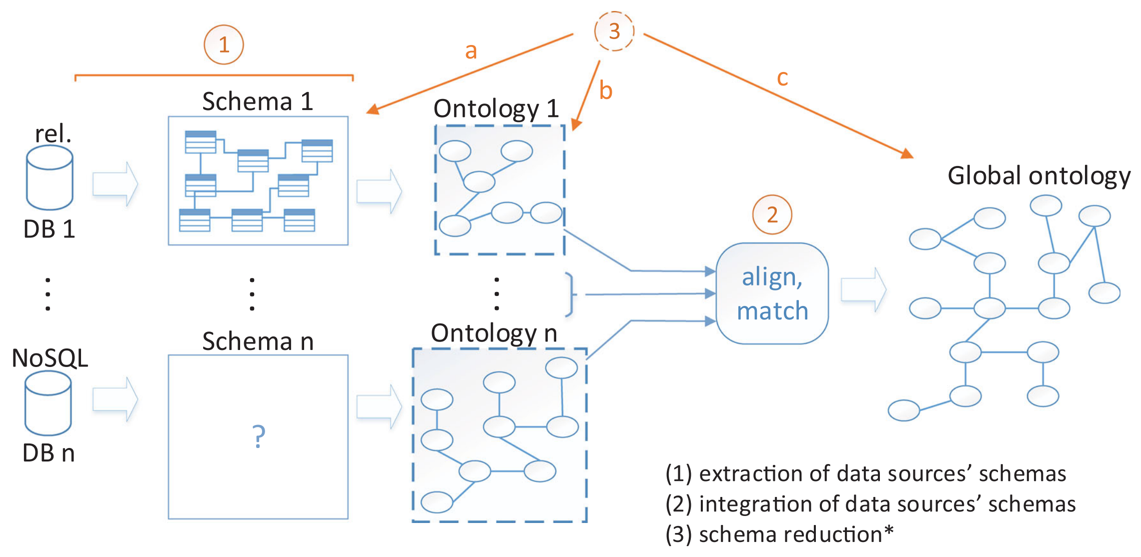
4.4. Automatization of design process

Automatization of DWH modelling phases was researched in the past, as well as now [11,14–16]. It reduces human, error-prone involvement and improves the accuracy and efficiency of repetitive tasks. Depending on the use case, data warehouse design steps can be automatized to a certain level.

Automatization is necessary for the integration of NoSQL DBs and DWH, but its complexity depends on the complexity of the data that are being integrated. Because of that, some modelling phases will have to be supervised. Therefore, full automatization is hardly possible and it is realistic to expect semi-automatization at most. More discussion on this matter is provided in Section 6.

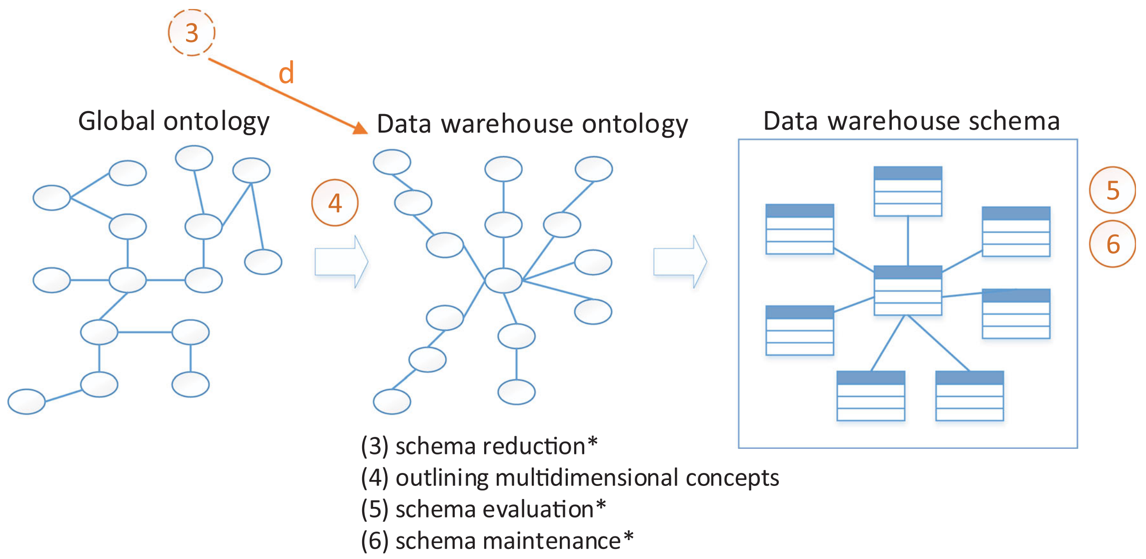
5. Semantics in data integration and data warehouse design

This section reviews research on the application of semantics in data integration (in general) and data warehousing – regardless of the involved data models (relational, XML, semantic, NoSQL) – that we find most important for our analysis of semantic-aided warehousing of NoSQL data. Review is organized by the most common flow of DWH design steps, similar to those in Figure 1. However, considering the integration of a NoSQL DB into a DWH, the *new* flow might be somewhat different; some steps might become iterative in



*Phase highlighted by an asterisk is either not necessarily applied in order specified by the list or is optional.

Figure 4. Design steps of ontology-based data integration/Steps of semantic-aided DWH design (part 1).



*Phases highlighted by an asterisk are either not necessarily applied in order specified by the list or are optional.

Figure 5. Steps of semantic-aided DWH design (part 2).

case of schema adjustment, but in general, steps remain the same – the complexity of their implementation varies. The *new* flow that could involve ontologies is shown in Figures 4 and 5

Figure 4 shows design steps that are common for both semantic-based data integration and the initial phase of semantic-aided DWH design: data sources are semantically integrated by matching their local ontologies (LOs) into a global ontology (GO), an ontological representation of the *integrated schema* from Figure 1. Applying it in the integration of data storages, GO acts as a virtualized unified view, enabling unified access to various storages. In the case of semantic-aided DWH design, GO is an entry point to further design phases, shown in Figure 5.

Figure 5 shows the remaining steps of semantic-aided DWH design, in which a DWH ontology is derived from the GO. It is done by gradual reduction and remodelling of the GO, and recognition and organization of MDCs – facts, dimensions and hierarchies into a DWH ontology, from which a DWH schema can be derived.

5.1. *Extraction of data sources' schemas*

5.1.1. *Extraction of the native schema*

The complexity of schema extraction depends on the data source type: for an RDB, it is a simple action of querying the system tables for the metadata; for a NoSQL data source, the schema has to be extrapolated or re-engineered from heterogeneous and almost *schemaless* data. However, document stores can be considered to have a schema – embedded and mixed with data – whose concepts are *implicitly* defined in relations of *containment*, *hierarchy* and *nesting* among collections, documents, *keys* and aggregations. Consequently, another problem is how to represent and store such schema. Research in this area involved tree-based schema representation and management [17,18], dictionary of concepts [19], and concept lattice [20].

Heterogeneity issues (Section 4.1) must be dealt with during this step, otherwise, the schema is ambiguous, inconsistent, and unclear. In the past research starting from the *relational era*, heterogeneities were manually or semi-automatically resolved by linguistic- and structure-based methods [21] and ontologies [22], or the combination of the two [12]. We believe a similar approach – that must not be fully manual – can be applied to document stores, and therefore we address *semantification* of data sources in further text.

5.1.2. Semantification of data sources

Ontologies help resolve heterogeneity within a data source or among multiple data sources, and they can also be used as their schematic representation and make heterogeneous stores *schematically comparable*. Past research resulted in solutions for ontology extraction from then popular storage types (e.g. text [23], RDBs [24], and XML [25]). Current research is mostly focused on NoSQL document stores [20,26–30]. In [29], a process of giving big data a virtual layer of high-level metadata description is called the *semantification* of big data. However, schema heterogeneity problems are also present during ontology derivation, making the fully manual approach unsuitable. Therefore, the idea of (semi-)automatized approach is also present in the current research.

5.2. Integration of data sources' schemas

If data sources' schemas are represented by an ontology, their integration becomes ontology matching and alignment problem. This field alone is a separate research area that yielded many algorithms and tools during the popularity of the semantic web in the penultimate decade, e.g. OLA [31], AROMA [32], FaCT++ [33], COMA++ [34], Pellet [35], etc. Some of them (e.g. Pellet and FaCT++) are nowadays commonly used and available as plugins in *Protégé*, one of the most popular ontology editing tool. The research in the ontology matching and reasoning field has been less intensive during the last decade but is still not at halt; many of these algorithms are still being improved by new ones (e.g. MOOM [30], CroMatcher [36], etc.). However, a DWH designer does not necessarily need expertise in ontology matching – which could rather be considered as a *black box* – but should have some insight on the features of the applied matching algorithm (e.g. whether they are structure-based or lexically based) because the quality of the output (GO) depends on it.

5.3. Outlining MDCs

Outlining of MDCs was done manually at the beginning of DWH era by the guidelines proposed by two main research currents [37,38]. Later on, new modelling methods were introduced, like e.g. *multidimensional fact* model [39], which utilized new graphical notation (i.e. *attribute tree*) that facilitated the design due to the simplicity of basic graph transformations. Later research also focused on integration of newly emerging data types (e.g. XML [40], linked data [41]), as well as overall (semi-)automatization of the design process [14,15,21,42,43]. The following list addresses the most common features in MDC discovery found in the past research of either (*Big*)DWH design attempts, or semantic-based DWH design, which we believe to have potential in warehousing of big data:

(1) *Utilization of functional dependencies (FDs) for dimension detection* – Since a fact-to-dimension relationship cardinality is n:1, *primary* and *foreign key* database integrity constraints and their cardinality have a vital role in determining which tables are facts and which contribute to the dimensions during the DWH design.
(2) *Discovery of functional dependencies (FDs)* – If the database schema lacks integrity definitions or if data lack schema in general, this can be compensated via data analysis. However, state of data current to time of analysis can only discover those FDs that *may* hold (at that time) *n:1* cardinality but can help eliminate with *certainty* those FDs that do not hold. After the discovery, FDs can be utilized in the way described in 1).
(3) *Heuristic measures for fact detection* [43] – Potential facts can be detected by using sets of heuristics (e.g. ratio of numerical attributes, number of attributes and instances, frequency of changes, relationship cardinalities, etc.), and then for each fact its dimensions can be detected using functional dependencies derived from schema definitions or the data as described in (1) and (2).
(4) *Demand-driven definition* – Users' requirements define the needed concepts. However, that requires a clear statement of the analysis scenarios and clear insight in the concepts available in the database.

Table 1 categorizes reviewed research by the mentioned features (column *MDC discovery*).

5.4. Schema reduction

Heterogeneity found in NoSQL data source can cause its schema to be large, but not necessarily having many analytically interesting concepts. Schema reduction has an important role in the integration of such source; applying user requests can help size down the

Table 1. Categorization of related DWH design research.

				Schema reduction		
References	Features *big* data	Features ontology	MDC discovery	Requirement statement	Reduction timing	Other features
[11,44]	+	+	4, 1 (based on [43])	*i** framework	Incrementally	model in attribute tree
[12]	−	+	3, 2, 1	NL, normalization	Early (on LOs)	tagged ontology
[15]	−	+	−	*i** framework, requirement ontology	Early (on LOs)	−
[13]	−	+	3, 1	NL, normalization	−	Requirement star schema
[14]	−	+	4	Requirement ontology	Early (on LOs)	
[43]	−	−	3, 1	−	−	−
[45]	−	+	1, 2, 3 (+4)	−	Incrementally	−
[41]	−	+	4	Requirement ontology	On GO	Semantic DWH

schema, thus in DWH modelling, the requirement-driven approach would have to be at least partially adopted, e.g. in form of *hybrid* approach as it was done in research [10–13,15,44].

Methods in the reviewed research differ in the way requirements were stated. Expressing requirements in a formalized format enhances automatization abilities of the modelling process. In [12,13], requirements were stated in a natural language format and then converted to a logical format (normalized). For the formalization of requirements, formalization frameworks like, e.g. *i** [11,15] can be used, as well as description logic [28]. Requirements can also be described by semantic resources [15], meaning that a *requirement ontology* is created [14,15] and can be included in the ontology matching process.

Other differences in the reviewed methods of major concern are regarding the *timing*, i.e. in which design phases are schema reduced by the application of the requirements. Applying requirements to the very data sources (their schema or LO) is considered an *early reconciliation* and in [12,14,15] is highlighted as a good approach. Besides on data sources' schemas or LOs, requirements can be applied after each schema representation is formed, meaning also on GO and DWH ontology as highlighted in Figures 4 and 5. Another option is incremental integration of conceptual schemas created on requirement- and data basis [11,44].

5.5. *Post-schema modelling phases*

After modelling the initial DWH schema, but before the ETL process, schema evaluation should take place. This can be done based on three criteria [46]: *disjointness*, *orthogonality*, and *summarizability*. Apart from the schema, ontology – either global or DWH ontology – can be evaluated terminologically [30] against the concepts from the domain of interest. Also, the whole design process could be evaluated for efficiency based on costs and manual involvement [47] or by evaluation metrics [48], which are mostly based on criteria described in Section 4 as main integration concerns.

Schema maintenance strategy is another important consideration. Changes on the NoSQL data source models are very likely to occur, exposing new analytically interesting data that could also cause changes in user requirements. That implies potential changes on the DWH schema for which a handling strategy must be chosen. Schema evolution is showing as a most promising [15,47]. DWH ontology could be maintained as its high-level presentation alongside the schema. If a multistore approach is chosen on a physical level, ontology could be used for mediated querying alongside a unified query language such as *bridge query language* [28] or *generalized query functions* [49].

6. Discussion

In this section, we provide the discussion and summary of reviewed topics and problems we find most important for the big data warehousing context.

6.1. *V-features of big data*

For warehousing of *big data*, expecting one solution for all problematic *V-features* is not realistic and they should be rather dealt with one at the time. These solutions seemingly depend on new technologies, but it is not generally applicable to all *V-features*.

Volume problem solutions strongly rely on the storage technology.

Solutions for *velocity* problem mostly depend on the processing technology. Also, it can be noticed that this problem bears resemblance to the real-time data warehousing problems – only on a significantly larger scale – and the topic of data stream warehousing should be researched further.

Veracity problem is also present in traditional data warehousing and is dealt with during the ETL phase in the data cleansing process. Predictably, it will most likely be dealt with at the very same phase in the *big* data warehousing.

We consider that the biggest problem currently for big data warehousing is the big data's *variety* feature because it imposes hurdles at the very first (modelling) stage of data warehouse creation and that therefore, *variety* should be dealt with foremost.

6.2. Role of ontologies in DWH design

Application of semantics, namely ontologies has potential in dealing with the *variety* problem. Ontologies can represent data sources' schema. Method of their extraction (especially from a NoSQL data source) is an active research question and it is yet to be researched whether they should be directly extracted from the source or from intermediary structures representing the schema.

Ontologies could also be used to find the missing data (demanded by requirements but not available in data sources) or to find analytically interesting concepts (not demanded by requirements but available in the data sources), which is discussed in more detail as *exploratory* approach in [50].

Once obtained, DWH ontology could have the role of virtual schema if, e.g. multistore approach is chosen on the physical level, and the queries would be mediated through the ontology acting as the integrated view of the data storages. Besides that, the benefit of having initial DWH ontology is the pre-definition of the domain of interest, inherited from the initial DWH that was already created within that domain. That means that the essence of initial user requests – already incorporated in the initial DWH – is inherently contained in both its schema and the ontology derived from it. Additional user requests could then be applied only to the new sources involved in modelling. The DWH ontology is a high-level abstraction of a star schema and would implicitly shape a data cube. For that reason, RDF Data Cube Vocabulary [51] and its concepts could be considered as a meta-model for a DWH ontology, although it was intended for the semantic (RDF-based) DWHs.

The role of semantics was also considered in post-modelling DWH implementation steps, such as ETL process. Research in [15] advocates ETL process should be *semantified* as well. If schema evolution took place, it would imply evolution of ETL process in parallel, and semantics could facilitate this evolution by introducing some form of automatization to it. For this reason, further research on *semantic ETL* and its automatization [11] is expected, but this is beyond the scope of this paper because the design is currently in our focus, while ETL is left for future research.

6.3. DWH modelling approach

BigDWH design should include any means possible for the reduction of extrapolated NoSQL data source schema, most promising of which are user requirements. Such design can originate from user requirements (requirement-driven approach) or both requirement- and data-driven strategies can take place parallelly (hybrid approach). Since it is unlikely there will be a clear *apriori* statement of DWH workload for the NoSQL scenario, hybrid approach is more suitable for the NoSQL integration coupled with *exploration* of the concepts available at the sources.

6.4. Automatization of DWH modelling

By definition, ontologies are *formal*, i.e. readable and processable by a computer, which is a prerequisite for automatization. Automatization of ontology matching and alignment, as well as reasoning from ontologies could bring further automatization possibilities to the semantic-based integration of a NoSQL data source with an existing DWH. The very data source schema or local ontology extraction can be automated to a certain degree, as well as the application of user requirements if they are formally stated. Previous research has been also focused on (semi-)automatization of extraction of MDCs from ontologies. Fully manual approach to any of these phases is questionable because it is error-prone and time-consuming. This all means *BigDWH* design steps could and should be all automated to a certain degree [16]. Considering the complexity of modelling imposed by NoSQL and big data problematic features, it is realistic to expect semi-automatization at most: the design process needs designer's supervision due to these features and questionable usefulness of the data, especially during extrapolation of ontologies.

6.5. Applying user requirements in DWH modelling

Choosing elements of hybrid approach means user requirements have to be included in the modelling process. The strategy of stating them and the timing of their application is another research matter. Using the formalized statement of requirements (also possible as ontologies) would enhance automatization of the design process that is needed to the greatest extent possible. It is potentially the most suitable form of stating them. If the DWH model becomes *evolutionary* and its design process becomes *iterative*, that suggests that *timing* of the requirement application should also be iterative, which is also yet to be researched.

6.6. Multidimensional design of BigDWH

Detection of MDCs in the RDBs was based both on sources' schema and data analysis, i.e. combination of reasoning from functional dependencies and heuristics. Similar approaches were attempted for the *BigDWH* design [11,44]. The most promising attempts to solving that problem included ontologies and we believe this idea should be pursued further in combination with the heuristic approach.

Solutions intended for the RDBs should be researched for an upgrade by applicability to new types of data. Similar research was also done on integration of XML files with the DWH [25,40]. Since XML format

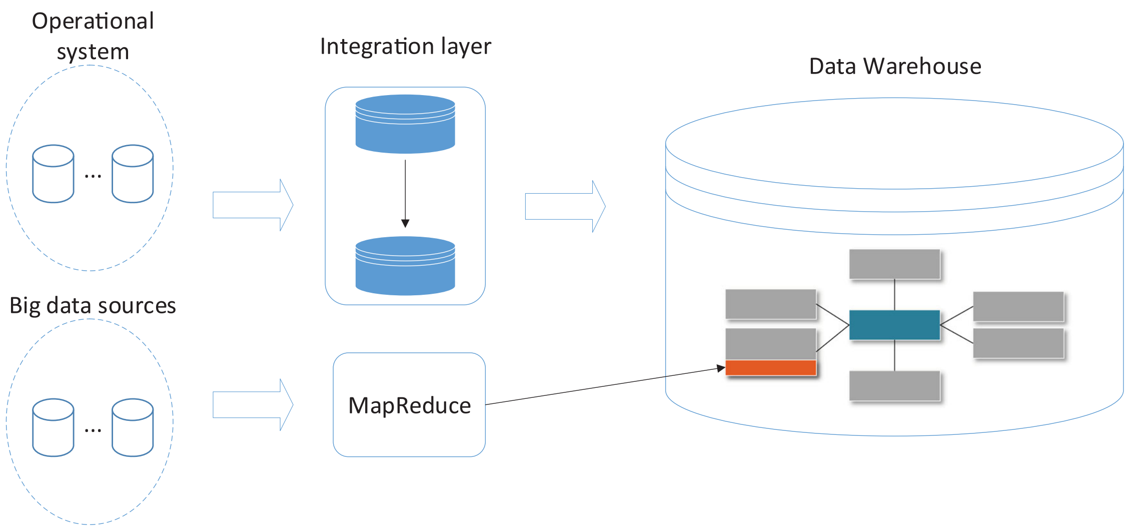


Figure 6. NoSQL DB *augments* existing DWH with additional dimensional attribute(s) [52].

bears resemblance with document stores (Section 3), these approaches should also be considered for an implementation in NoSQL and JSON environment, targeted for the *BigDWH* integration.

Additionally, multiple scenarios must be considered and supported regarding the DWH concept big data could end up as by design. We identify three basic scenarios of new MDC created by big data:

(1) new attributes (dimensional, or measures) (Figure 6)
(2) new dimensions
(3) new fact table.

Each of these cases has its own particularities and sub-scenarios, e.g. an aggregation of portion of a big data set could be the newly generated attribute for the first case, which is, e.g. a question of granularity.

6.7. Overall design of BigDWH

In the reviewed ontology-based integration scenarios, it is noticeable that all approaches to DWH design are *from scratch* – at the entry point, there are only *raw* data sources. The applicability of this approach in practice should be reconsidered, considering the current maturity of data warehousing. We argue that a more realistic starting point is having a traditional relational DWH and a new (NoSQL) data source that is intended as its upgrade, transforming it in a way to a *BigDWH*.

According to this design *shift*, "rel. DB 1" in Figure 4 would be replaced by a relational DWH as the first input into the initial design phase, and its schema extraction (corresponding to "Schema 1" in Figure 4) should not be problematic. "Ontology 1" would become an (initial) DWH ontology, different from the DWH ontology in Figure 5 which is its *extension*, made by its matching with the ontology of the new data source.

New (NoSQL) data source is the other input into the initial design phase. Currently, we consider (adding) one new data source at the time due to the potential complexity and leave considerations of multiple data sources for future research. In the future, ontologies could be used to discover reciprocal replenishment of the involved data sources and could be aligned all at once, one by one, or with the DWH ontology.

Modality of interaction with the NoSQL data sources should also be considered in terms of their *independence*. Authors in [30] advocate it, meaning the only interaction is copying them. This matter has a greater weight if the data source is highly characterizable by *velocity* and *variety* in the structure of incoming data because that could have an effect on the *schema*. Advocated *independence approach* could be an acceptable strategy if compromises are made. A *snapshot state* of the data source could be taken, implicitly containing its *schematic snapshot*, which would substitute the intensive interaction with the NoSQL data sources onto their *copies*. In case of very voluminous NoSQL data source, parts of it could be copied for schema sampling. The compromise made with this approach is the data recentness as well as *schematic recentness*. All of the above needs to be researched and elaborated further.

7. Conclusion

This paper discusses problems of integration of traditional data warehouses and NoSQL databases from the point of view of general integration, as well as data warehouse design, and reviews the potential role of ontologies in overcoming them. Ontology-based approaches to data integration and data warehouse design phases of various data sources are reviewed and future research directions are discussed, all aimed at alleviation of data warehouse design empowered by new kinds of data.

Designers must be provided an exploratory tool that would facilitate inspection of new NoSQL data sources and they need as much automatized aid in design as possible, e.g. recognition and suggestion of multidimensional concepts available from NoSQL data sources. One of the most promising approaches is using ontologies in combination with methods and heuristics developed with the same purpose for the relational data sources. Heuristics can aid in detecting relationships between concepts directly from the data, and the ontologies can help in capturing the semantical essence of the domain of analytical interest. Most importantly, their automatized reasoning enhances higher automatization degree in various data warehouse design stages, e.g. resolving heterogeneity conflicts, discovering hidden semantics, detection of multidimensional concepts and – most importantly – exploring large volumes of unstructured data, which cannot be done manually. However, due to the specifics of big data, the design should be supervised due to conflicts that need to be resolved manually, thus we believe that expectations on automatization must remain realistic and that semi-automatization would be the best choice.

To the best of our knowledge, research done in this field approached *big data warehouse* design starting from scratch, but we believe *big data warehouse* should be a redesign of an existing data warehouse, starting the (re-)design by having a data warehouse to which a new, NoSQL data source intends to be added. Developing this idea, data warehouse schema maintenance strategy must be researched further, possibly for adaptation of schema evolution approach, as well as utilization of ontologies as schematic representations as they are easily updateable.

Research in the integration of NoSQL databases with data warehouse was so far mostly focused on document store family and we believe it should be researched further in the mentioned context of semantic-based integration. Most likely NoSQL family to be researched next are column stores, which have been put in focus alongside document stores in the recent research. Predictably, further research on applying approaches to semantically enhanced integration of NoSQL databases and data warehouses – similar to the ones being researched for document stores – can be expected for column stores and the rest of the NoSQL database families.

Note

1. Most popular NoSQL database engine according to various popular data storage-devoted websites' statistics (e.g. *db-engines.com*).

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This research has been supported by the European Regional Development Fund under the grant KK.01.1.1.01.0009 (DATACROSS).

ORCID

Marina Ptiček http://orcid.org/0000-0002-3186-7390
Boris Vrdoljak http://orcid.org/0000-0003-0081-172X
Marko Gulić http://orcid.org/0000-0002-6637-2582

References

[1] Ziegler P, Dittrich KR. Three decades of data integration — all problems solved? In: Jacquart R, editor. Building the information society. Boston (MA): Springer US; 2004. p. 3–12.
[2] Bowermaster P, Powell R. Is It Time to Re-Think Data Modeling? [Internet]. BeyeNETWORK. 2018 [cited 2019 Jan 20]. Available from: http://www.b-eye-network.com/view/17505.
[3] Gacitua R, Mazon JN, Cravero A. Using semantic web technologies in the development of data warehouses: a systematic mapping. WIREs Data Mining Knowl Discov 2018. doi:10.1002/widm.1293
[4] Pticek M, Vrdoljak B. Semantic web technologies and big data warehousing. 41st International convention on information and Communication technology, Electronics and Microelectronics (MIPRO). Opatija (Croatia): IEEE; 2018. p. 1214–1219.
[5] Thakur G, Gosain A. DWEVOLVE: a requirement based framework for data warehouse evolution. ACM SIGSOFT Software Engineering Notes. 2011;36:1–8.
[6] Golfarelli M, Lechtenbörger J, Rizzi S, et al. Schema versioning in data warehouses: enabling cross-version querying via schema augmentation. Data Knowl Eng. 2006;59:435–459.
[7] Shvaiko P, Euzenat J. A survey of schema-based matching approaches. In: Spaccapietra S, editor. Journal on data semantics IV. Berlin: Springer Berlin Heidelberg; 2005. p. 146–171.
[8] Ptiček M, Vrdoljak B. Big Data and New Data Warehousing Approaches. Proceedings of the 2017 International Conference on Cloud and Big Data Computing (ICCBDC). London: ACM Press; 2017. p. 6–10.
[9] Chen Y, Xu C, Rao W, et al. Octopus: hybrid big data integration engine. Vancouver: IEEE; 2015. p. 462–466.
[10] Di Tria F, Lefons E, Tangorra F. Academic data warehouse design using a hybrid methodology. Comput Sci Inf Syst. 2015;12:135–160.
[11] Di Tria F, Lefons E, Tangorra F. Big data warehouse automatic design methodology. Big data management, technologies, and applications. IGI Global. 2014: 115–149. doi:10.4018/978-1-4666-4699-5.ch006.
[12] Thenmozhi M, Vivekanandan K. An ontology based hybrid approach to derive multidimensional schema for data warehouse. Int J Comput Appl. 2012;54: 36–42.
[13] Elamin E, Feki J. Toward an ontology based approach for data warehousing state of the art and proposal. Proceedings of the International Arab Conference on Information Technology (ACIT2014). Nizwa: University of Nizwa; 2014. p. 170–179.
[14] Aadil B, Wakrime AA, Kzaz L, et al. Automating data warehouse design using ontology. Tangiers: IEEE; 2016. p. 42–48.
[15] Thenmozhi M, Vivekanandan K, Gayathree G. A semi-automatic approach to handle data warehouse schema evolution using ontology. Proceedings of Fourth International Joint Conference AET 2013. Gurgaon: Elsevier; 2014. p. 55–56.

[16] Romero O. Automating the multidimensional design of data warehouses. Barcelona: Universitat Polit‘ecnica de Catalunya; 2010.
[17] Wang L, Zhang S, Shi J, et al. Schema management for document stores. Proceedings of the VLDB Endowment. 2015;8:922–933.
[18] Gallinucci E, Golfarelli M, Rizzi S. Schema Profiling of Document Stores. Proceedings of the 25th Italian Symposium on Advanced Database Systems. Squillace Lido (Catanzaro), Italy: CEUR-WS.org; 2017. Available from: http://ceur-ws.org/Vol-2037/paper_3.pdf.
[19] Hamadou HB, Ghozzi F, Peninou A, et al. Querying heterogeneous document stores. Funchal: SciTePress; 2018. doi:10.5220/0006777800580068.
[20] Jabbari S, Stoffel K. Ontology extraction from MongoDB using formal concept analysis. 2nd International Conference on Knowledge Engineering and Applications (ICKEA). London: IEEE; 2017. p. 178–182.
[21] Banek M, Vrdoljak B, Tjoa AM, et al. Automating the schema matching process for heterogeneous data warehouses. In: Song IY, Eder J, Nguyen TM, editors. Data warehousing and knowledge discovery. Berlin: Springer Berlin Heidelberg; 2007. p. 45–54.
[22] Ram S, Park J. Semantic conflict resolution ontology (SCROL): An ontology for detecting and resolving data and schema-level semantic conflicts. IEEE Trans Knowl Data Eng. 2004;16:189–202.
[23] Onto LT. A Protégé plug-in for ontology extraction from text. 2003.
[24] Cerbah F. Learning highly structured semantic repositories from relational databases: The RDBToOnto tool. In: Bechhofer S, Hauswirth M, Hoffmann J, et al., editors. The semantic web: research and applications. Berlin: Springer Berlin Heidelberg; 2008. p. 777–781.
[25] Rodrigues T, Rosa P, Cardoso J. Moving from syntactic to semantic organizations using JXML2OWL. Comput Ind. 2008;59:808–819.
[26] Abbes H, Gargouri F. M2Onto: An approach and a tool to Learn OWL ontology from MongoDB database. In: Madureira AM, Abraham A, Gamboa D, et al., editors. Intelligent systems design and applications. Cham: Springer International Publishing; 2017. p. 612–621.
[27] Kiran VK, Vijayakumar R. Ontology based data integration of NoSQL datastores. Gwalior: IEEE; 2014. p. 1–6.
[28] Curé O, Lamolle M, Le Duc C. Ontology based data integration over document and column family oriented NOSQL stores. Bonn: ArXiv; 2011.
[29] Mami MN, Scerri S, Auer S, et al. Towards semantification of Big data technology. In: Madria S, Hara T, editors. Big data analytics and knowledge discovery. Cham: Springer International Publishing; 2016. p. 376–390.
[30] Abbes H, Gargouri F. MongoDB-based modular ontology building for big data integration. J Data Semant. 2017; 7: 1–27.
[31] Euzenat J, Valtchev P. Similarity-based ontology alignment in OWL-Lite. Proc. 16th European conference on artificial intelligence (ECAI). Valencia; 2004. p. 333–337.
[32] David J, Guillet F, Briand H. Matching directories and OWL ontologies with AROMA. Arlington (VA): ACM Press; 2006. p. 830–831.
[33] Tsarkov D, Horrocks I. FaCT++ description logic reasoner: system description. Proceedings of IJCAR 2006. Springer; 2006. p. 292–297.
[34] Aumueller D, Do H-H, Massmann S, et al. Schema and ontology matching with COMA++. Baltimore (MD): ACM Press; 2005. p. 906–908.
[35] Sirin E, Parsia B, Grau BC, et al. Pellet: A practical OWL-DL reasoner. J Web Semant. 2007;5:51–53.
[36] Gulić M, Vrdoljak B, Banek M. Cromatcher: an ontology matching system based on automated weighted aggregation and iterative final alignment. J Web Semant. 2016;41:50–71.
[37] Kimball R, Ross M. The data warehouse toolkit: the definitive guide to dimensional modeling. 3rd ed. Indianapolis (IN): Wiley; 2013.
[38] Inmon WH. Building the data warehouse. 3rd ed. New York: Wiley; 2002.
[39] Golfarelli M, Maio D, Rizzi S. The dimensional fact model: a conceptual model for data warehouses. Int J Coop Inf Syst . 1998;07:215–247.
[40] Vrdoljak B, Banek M, Skočir Z. Integrating XML sources into a data warehouse. In: Lee J, Shim J, Lee S, et al., editors. Data engineering issues in E-Commerce and services. Berlin: Springer Berlin Heidelberg; 2006. p. 133–142.
[41] Bellatreche L, Khouri S, Berkani N. Semantic data warehouse design: from ETL to deployment à la Carte. In: Meng W, Feng L, Bressan S, et al., editors. Database systems for advanced applications. Berlin: Springer Berlin Heidelberg; 2013. p. 64–83.
[42] Banek M, Vrdoljak B, Tjoa AM, et al. Automated integration of heterogeneous data warehouse schemas. Int J Data Warehouse Min. 2008;4:1–21.
[43] Carmè A, Mazón J-N, Rizzi S. A model-driven heuristic approach for detecting multidimensional facts in relational data sources. In: B Pedersen T, Mohania MK, Tjoa AM, editors. Data warehousing and knowledge discovery. Berlin: Springer Berlin Heidelberg; 2010. p. 13–24.
[44] Di Tria F, Lefons E, Tangorra F. Design process for big data warehouses. Shanghai: IEEE; 2014. p. 512–518.
[45] Romero O, Abelló A. Automating multidimensional design from ontologies. Proceedings of the ACM tenth international workshop on Data warehousing and OLAP (DOLAP ‘07). Lisbon: ACM Press; 2007. p. 1–8.
[46] Nebot V, Berlanga R, Pérez JM, et al. Multidimensional integrated ontologies: a framework for designing semantic data warehouses. In: Spaccapietra S, Zimányi E, Song I-Y, editors. Journal on data semantics XIII. Berlin: Springer Berlin Heidelberg; 2009. p. 1–36.
[47] Thenmozhi M, Vivekanandan K. An ontological approach to handle multidimensional schema evolution for data warehouse. Int J Database Manag Syst. 2014;6:33–52.
[48] Di Tria F, Lefons E, Tangorra F. Evaluation of data warehouse design methodologies in the context of big data. In: Bellatreche L, Chakravarthy S, editors. Lyon: Springer International Publishing; 2017. p. 3–18.
[49] Vathy-Fogarassy Á, Hugyák T. Uniform data access platform for SQL and NoSQL database systems. Inf Syst. 2017;69:93–105.
[50] Abello A, Romero O, Pedersen TB, et al. Using semantic web technologies for exploratory OLAP: a survey. IEEE Trans Knowl Data Eng. 2015;27:571–588.
[51] The RDF Data Cube Vocabulary [Internet]. W3C Recommendation; 2014. Available from: https://www.w3.org/TR/vocab-data-cube/.
[52] Jukić N, Sharma A, Nestorov S, et al. Augmenting data warehouses with big data. Inf Syst Manage. 2015;32:200–209.