Formalization of materialized view problem in ontology-based databases

Bery Leouro Mbaiossoum[1], Narkoy Batouma[1], Atteib Doutoum Mahamat[1], Ouchar Cherif Ali[2], Lang Dionlar[1], Ladjel Bellatreche[3]
[1]Department of Computer Science, Faculty of Exact and Applied Sciences, University of Ndjamena, N'Djamena, Chad
[2]Department of Computer Science, Institut National Supérieur des Sciences et Technique d'Abéché, Abéché, Chad
[3]LIAS, École Nationale Supérieure de Mécanique et d'Aérotechnique (ENSMA), Poitiers, France

Article Info

Article history:

Received Oct 16, 2024
Revised Jul 19, 2025
Accepted Oct 15, 2025

Keywords:

Materialized views
Ontology-based-databases
formalization
Semantic databases
SPARQL queries

ABSTRACT

Materialized views are essential for optimizing the performance of traditional databases and data warehouses by accelerating query responses. However, their substantial storage requirements and the impracticality of materializing all possible views raise the problem of selecting which views to persist, a fundamental physical design challenge. This article presents a rigorous formalization of this problem within the context of semantic databases. The methodology employed includes a comprehensive literature review aimed at identifying the variety of se-mantic database representations. This analysis revealed a significant diversity in data models and query languages used. Based on this analysis, a generic formalization framework is pro-posed. This framework enables the expression of various resolution approaches to the materialized view selection problem, taking into account the specificities of semantic databases. It offers broad applicability to any database management system, providing a common language to describe and compare view selection methods.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Bery Leouro Mbaiossoum
Department of Computer Science, Faculty of Exact and Applied Sciences, University of Ndjamena
Route de Farcha, Ndjamena, Chad
Email: bery.mbaiossoum@gmail.com

1. INTRODUCTION

Query optimization is an important issue in the context of databases. She has always had an important place across the different generations of databases: traditional databases, XML databases, data warehouses, statistical and scientific databases, and semantic databases. Since database applications are always looking for more efficient query processing times, several works have been carried out to make query optimizers more efficient. In traditional databases, several query optimization techniques have been proposed and some have given good results. We can cite for instance, the techniques of indexing, clustering, fragmentation, and materialized views.

Materialized views are one of the most successful optimization techniques in relational databases (RDB) [1] and in data warehouses [2]-[5]. Indeed, materialized views make it possible to respond efficiently to queries, especially in terms of execution time. However, materialized views occupy enough storage space that it is difficult to materialize all the possible views. It is then necessary to select some of them for materialization. View selection truly emerged as a physical design problem with the development of data warehouses [2], [6].

In semantic databases, little work on optimizing queries using materialized views has been carried out. And among the works carried out [7]-[9], very few really take into account the semantic dimension (ontological dimension) of these databases. The databases are considered as tables of triples. But these databases can adopt several representation schemes. In addition, these works deal with RDF type data, while semantic databases use other data formalisms such as OWL [10], and PLib [11].

In this work, we will focus on materialized views for the optimization of semantic databases. We will consider the diversities presented by semantic databases. As in traditional databases, the problem is that of selecting views to persist. Then, we will present the problem of selecting materialized views and we will propose a formalization of the problem which will make it possible to circumscribe the different approaches to solving this problem. And we will propose a resolution schema of this problem.

As a plan for this work, we first present the problem of view selection in classical databases and in semantic databases. We will propose, secondly, a formalization of the view selection problem in semantic databases. We will define the different components of this formalization, notably the set of views in the different types of semantic databases. Next, we will discuss the resolution schema of this problem. Finally, we will discuss the characterizations of the queries and a conclusion will follow.

2. METHOD

To tackle this work, we perform a literature review and an analysis of the research studies focusing on materialized views in order to discover the main components of materialized views problem to formalize. After that, based on the observed materialized views problem solving approaches, we propose a global resolution schema of this problem. So, we start by stating the materialized views problem in different types of databases.

2.1. Materialized view selection problem in traditional databases

In RDB and data warehouses, the materialized view selection problem is formulated as: given Q={q1, q2, ..., qn}, a set of queries, DBS, a schema database i.e., DBS ={R1, R2, ..., Rk} where the Ri are tables; and some constraints (storage space, view maintenance cost, and number of views). The objective function consists of finding a set of views which optimizes the queries of Q and which respects the constraints [2]. The database schema is known. This is generally the relational model. The entire resolution process is based on it.

2.2. Materialized view selection problem in semantic databases

In semantic databases, there are several ways to represent ontologies and their instances. The commonly used representations are: the vertical representation [12]-[17] and the hybrid representation. So, the database schema is not always known in advance. The problem of selecting materialized views is generically presented as: given an ontology and its population, a set of queries Q={q1, q2, ..., qn} and some constraints (e.g. space of storage, view maintenance cost, number of views, and response time). The objective function consists of finding a set of views which minimizes the processing of Q queries and which respects the constraints. As the schema of the representation of the semantic database is therefore not immediately known, this gives rise to two approaches for solving this problem:

– Either we target a representation of ontologies (schema and instances) and we base all the reasoning on the latter, we speak of targeted views. Most of the methods encountered for solving the materialized view selection problem (PSVM) use this approach,
– or we reason at the level of the ontology schema, then we adapt the results to an implementation at the time of production, we will talk about generic views.

However, we note that the formalization of this problem in the two types of databases (RDB and SDB) is almost identical. The main difference is that in the case of SDB, the representation schema of the ontology and that of its instances are not known. So, to be able to solve this problem in semantic databases, we need other information including ontology representation schemes and ontological instances. In other words, the formalization of the problem as presented is incomplete. We see this even in classic databases, mainly in data warehouses where there are different ways of implementing them [18]: relational OLAP (ROLAP), multidimensional OLAP (MOLAP), and hybrid OLAP (HOLAP). To make this formalization uniform for all types of databases, we propose to provide a characteristic dimension of the databases.

To do this, we will use the SDB formalization that we proposed in [19] which provides all the information on a given semantic database. We will focus on semantic databases but it is possible to adapt the work to the case of classic databases. Let us recall this formalization.

2.3. Formalization of semantic databases

With the growth of ontological instances, several database models that can support these instances have been proposed. These databases are called semantic databases (SDB) or ontology-based databases (OBDB). The development of numerous SDB results mainly from:

a) The diversity of formalisms: each OBDB uses a particular formalism to define its ontologies (RDF, OWL, PLIB or FLIGHT);

b) The diversity of storage models: unlike traditional databases, where the logical model is stored using a relational approach, in semantic databases, a variety of storage models (horizontal and specific representation) are used to store the two levels of modeling: ontology level and ontological instances level;
c) The diversity of target architectures used by the database management system: an OBDB can use a single or several database schemas to store all the data.

The diversities of the OBDB proposed make difficult the formalization of materialized view problems. Thus, we propose to use the following formalization of OBDB to facilitate the formalization of the problem of materialized views. This formalization of OBDB is an abstract structure allowing the diversity in OBDB to be represented in terms of: ontology model, ontological instances, ontological model storage schema, ontological instance storage schema and OBDB architecture. Then, let us recall the formalization of ontology model.

2.4. Formalization of the ontology model

An ontology model is generically formalized by the following 5-tuple: *<C, P, Applic, Ref, Formalism>* [20] where:
– *C* represents the classes of the ontology model;
– *P* represents all the properties of the ontology model;
– *Applic:* $C \rightarrow 2^P$ is a function which allows to link each class to the properties attached to it;
– *Ref:* $C \rightarrow$ *(operator, Exp(C))* is a function which associates each class with an operator (inclusion or equivalence) and an expression on other classes. The expressions defined for OWL ontologies based on description logics present, from our point of view, a complete set of operators covering several formalisms of ontologies. These expressions use the following operators: set operators (intersectionOf (…), unionOf (…), complementOf (C)), property restrictions (AllValuesFrom, SomeValuesFrom, HasValue) and cardinalities operators (~nR.C , <nR.C). *Exp* can be the identity function that associates a class with the same class (class defines itself as the largest class in the “Thing” hierarchy).
– *Formalism* is, as its name indicates, the formalism of the ontology model adopted.

For example, a PLIB ontology will be defined as an instance of the following 5-tuples: <Classes, Properties, Applic, OntoSub(…), PLIB>. Plib's OntoSub operator is an operator for defining partial inheritance, where a class references another class by inheriting all or part of its properties. An OWL ontology will be defined by: <Classes, Properties, Applic, Description Logic operators, OWL>. *Formalization of BDBOs.* We define a SDB formally as a 7-tuple: *SDB:< MO, I, Sch, Pop, SMMO, SMInst, Ar >, where:*
– *MO*: represents a generic ontology model *<C, P, Applic, Ref, Formalism>;*
– *I*: represents all ontology instances;
– *Sch:* $C \rightarrow 2^P$ is a function which associates with each class the set of properties for which the instances of this class are valued;
– *Pop:* $C \rightarrow 2^I$, is a function which associates each class with its instances, part of the set *I*;
– *SMMO*: the storage model which is storage schema of the ontology model (vertical and horizontal);
– *SMInst*: storage model which is storage schema for ontology instances.
– Architecture model (Ar): the architecture type of the database (Type I, II, or III).

According to this formalization, the SDB of Oracle, IBM SOR, and OntoDB are respectively represented by:
– SDB-Oracle: <MO:<Classes, Properties, Applic, Operators (RDFS, OWLSIF and OWL Prime), (RDFS, OWLSIF or OWLPrime depending on the version)>, RDF instances, φ, tables RDF_link and RDF_values tables giving the instances of each class, Vertical, Vertical, Type I>. The function φ associates with each class the set of properties for which the instances of this class are valued; φ can be a query on the database.
– SDB-IBM: <MO:<Classes, Properties, Applic (properties of each class), description logic operators, OWL>, Owl Instances, sch (query returning the valued properties for each class), Instances of each class, Horizontal, Binary, Type II>.
– SDB-OntoDB: <MO:<Classes, Properties, Applic (properties of each class), OntoSub operator, PLIB>, Plib instances, Sch $\subseteq$ Applic, Instances of each class, Horizontal, Horizontal, Type III>.

3. RESULTS AND DISCUSSION

3.1. Formalization of the materialized view selection problem

Having a database thus formalized, we have a good knowledge of it, in particular the storage schema of its ontology model, the storage schema of its instances and its implementation architecture. Then, we formalize the problem of materialized views as a 4-tuple: *<SDB, Q, C, V >* where
– *SDB: < MO, I, Sch, Pop, SMMO, SMInst, Ar >*, a semantic database;

– *Q*: a load of requests;
– *C*: constraints relating to the problem and;
– *V*: a set of possible views.

The problem therefore consists to find W⊆V such that W minimizes the treatment of the charge Q and satisfies the constraints C. With this model, we have good knowledge of the database in question and we can focus on searching W which is an optimal set of views. We will subsequently characterize the different components of this formalization, mainly the set of possible views V and the queries.

3.2. Possible views to be materialized

The set of possible views depends on the schema adopted for the representation of ontology instances. To do this, we will identify this set for each type of semantic database. We recall that there are three main types of SDB:

– SDB of type I (SDBI): these are the semantic databases which use the vertical approach consisting of storing the ontology and its instances in a table with three columns (subject, predicate, object) representing respectively: (i) the identifier of the ontological resource (class, property or ontological instance), (ii) the name of the resource, and (iii) the value of this resource. This representation has the advantage of facilitating the implementation and insertion of new triples. But querying it is complex because it may require several self-join operations. Semantic databases [12]-[14] use this representation.
– SBD of type II (SDB II): in this type, the approach used for the representation of ontology instances is the binary approach: it consists of decomposing the relationships into two categories: unary relationships (for class membership), and binary relationships (for property values). This binary approach comes in three variants depending on the approach adopted for the representation of inheritance: (i) a single table for all classes of the ontology, (ii) one table per class with table inheritance (if an object relational DBMS is used), and (iii) one table per class without table inheritance. This approach is used in SDB [15], [17].
– SDB of type III (SDB III): in this type of SDB, we use the horizontal approach which is similar to the traditional representation used by relational DBMS. It consists of associating with each ontology class a table having a column for each property associated with a value for at least one instance of this class. The SDB OntoDB [20] uses the horizontal approach for the representation of its ontology model and also of it instances.

3.2.1. Possible views in SDB II and SDB III

In these two types of semantic databases, we use relational or object-relational model with a set of tables. We use the notion of the universal relation in RDB to define the set of possible views. The universal relation is a relation that results from the natural joining of all relations in the database. It is a concept that is commonly used to express the semantics of dependencies in RDB. The fundamental idea is to relieve the user (or the user program) of the explanation of the data access paths, by providing him an interface which gives him the impression that the queries are made on a single relation. Let R(P1, P2,..., Pn) be the unique relation composed of all the attributes of the database. R is a subset of the Cartesian product of the database attributes Pi, U = P1 × P2 × ... × Pn. By grouping certain properties, we can decompose R into small relations.

R can have several decompositions, some of which carry the risk of loss of information. Frasincar *et al.* [21] showed that we can decompose R into several tables without loss of information by considering functional dependencies and join dependencies. Which amounts to placing the attributes in their respective tables if the dependencies exist. R becomes a series of joins of database tables. In other words, U is the Cartesian product of the relations Ri of the SDB (U=×Ri). We define a view as the result of an algebraic operation on the universal relation U. Then, the set of possible views is: $V = \{v \mid v \in U\}$, where U is a universal relation and v is a result of a query on U (i.e., a view).

3.2.2. Possible views in SDB I

In this type of semantic databases, we are dealing with a single table. The set of possible views can be defined in several ways aiming at the attitude we adopt towards the queries: we can define the set of possible views by considering either the SDB and the queries or the SDB alone.

A. Proposal considering the SDB and queries

We have *Q* a load of queries. To answer each query, a certain number of self-joins on the triple table are necessary. If we denote by n the maximum number of self-joins on the table of triples to answer all the queries of *Q*. This number n is less than or equal to the number of triples of the largest query of the load (because certain triples constitute filters and do not require a self-join of the triples table). Then, any view is a result of an algebraic operation on the Cartesian product T × T × ... × T, (n times), with T the table of triples. We define all of these views by: $V = \{v \mid v \in T^{\wedge}n\}$ where T^n is the nth Cartesian product of the table of triples, and n the maximum number of self-joins for Q indicated above.

B. Proposal considering only the SDB

In this approach, we only consider the table of triples. If we perform a horizontal representation of this table, we obtain a table whose columns are the properties of the ontology and the rows are the instances of the ontology. Figure 1 provides an illustration.

This representation is not optimal for the simple fact that it presents several null values. But it serves to express the set of possible views. Indeed, with this representation, and according to [22], the universal relation is the Cartesian product of all the attributes of the database: $U = P1 \times P2 \times ... \times Pn$ with the Pi the attributes of the horizontal relation (representation) of the table of triplets. The set of views is then defined by: $V = \{v | v \in RU = Id \times P1 \times P2 \times . . . \times Pn\}$.

C. Proposal taking into account the table of triples and the ontology schema

A semantic database type I uses the same schema for the representation of the ontology and for those of the instances. All data is stored in the same table in the form of triplets. It is possible to identify the reference class for each triplet. Thus, let I be the set of instances of the ontology (I = {i | i instance of C}). If we place each instance i in its membership class Ci, and if we extrapolate that a relation Ri corresponding to Ci exists, then we can apply the definition of the universal relation of [Ulman, Fagin] on this database. Thus, the set of views is defined as: $V= \{v| v \in U=R\_1 \times R\_2 \times ... \times R\_n\}$, v a result of a query on U (i.e., a view).

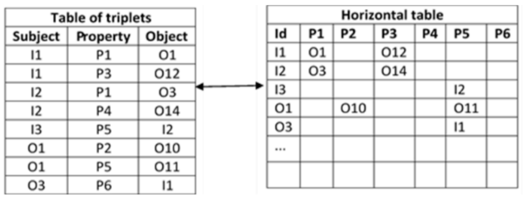


Table of triplets		
Subject	Property	Object
I1	P1	O1
I1	P3	O12
I2	P1	O3
I2	P4	O14
I3	P5	I2
O1	P2	O10
O1	P5	O11
O3	P6	I1

Horizontal table						
Id	P1	P2	P3	P4	P5	P6
I1	O1		O12			
I2	O3		O14			
I3					I2	
O1		O10			O11	
O3					I1	
...						

Figure 1. Tables of triplets and horizontal table corresponding

3.3. Resolution of the PSVM

The problem of selecting materialized views having thus been formulated, its resolution will consist of finding a function ϕ defined as: $\phi :<SDB, Q, C, V> \rightarrow W$ where $W \subseteq V$ and if f is a cost function, for all $W' \subseteq V, f(W) \leq f(W')$ and W satisfies the set of constraints C. The function ϕ represents any approach to solving the materialized views problem. In semantic databases, ϕ can represent for example the approach of Castillo and Leser [7]. In other words, the latter is an instantiation of ϕ. We therefore write: $W = \phi(SdbI, Q, S, V)$ with *SdbI: <MO:<Classes, Properties, Applic, Operators (RDFS), RDF>, RDF Instances, φ, Triplet Table, Vertical, Vertical, Type I>*. The *Applic* function can be a query which, for each class c, returns all the properties. In SPARQL, this query will look like: select ?p where (?p rdfs:domain c). Pop(c) which returns the instances of each class c, looks like: select ?i where (?i rdf:type c). $Q = \{\Lambda \text{ triplets}\}$, C: storage space, V: all views on the table of triplets.

The function ϕ must therefore be instantiated by all resolution approaches of materialized view problem. But unfortunately, in some approaches, query rewriting, which is an important phase in solving the materialized view problem, is processed simultaneously with the search for the optimal set of views. Then, there is a problem because ϕ does not take into account the rewriting of queries. For these approaches, we propose another function taking into account the rewriting of queries.

Let γ be this function. $\gamma :<BDBO, Q, C, V> \rightarrow <W, Re>$ where $W \subseteq V$ and $Re = \{r(q) \mid r(q): \text{rewriting of } q, q \in Q\}$, the set of queries rewritten. If f is a cost function, for all $W' \subseteq V, f(W) < f(W')$ and W satisfies the constraint set C. The function γ represents any approach dealing simultaneously with the selection and queries rewriting. For instance, [8] presents a materialized view problem resolution which can be represented as: $<W, R> = \gamma(SdbI, Q, S, V)$ with *SdbI:< MO:<Classes, Properties, Applic, Operators (RDFS), RDF>*, RDF instances, *φ*, Triples table, Vertical, Vertical, Type I>, $Q=\{\Lambda \text{ triples}\}$, V: all views on the triples table and $C = \{\text{processing cost, storage cost, cost of maintenance}\}$, W: optimal set of views and R: set of queries rewritten on W.

3.4. Characterization of queries

In most query languages, we can characterize queries by their types (select and update), the data source they relate to, and their results. In RDBs, SQL is the standard query language. The types of SQL queries are: selection, projection, join, modification (update), and set operations. The main data source is a set of tables i.e., a relational database (but we can have other data sources such as text files) and its results are a set of tuples. In object-oriented databases (OODB) or object-relational databases (ORDB), the standard languages are OQL and SQL. Their query types are those of RDB to which we add some method invocation and path expression operations (flatten, coalence). The data source is the set of tables and the results are either a collection (set, bag, array) of objects, literals, or an object or a literal. In SDB, the commonly used query language is the SPARQL language [21], [23], designated by the W3C consortium as the standard language for semantic queries. In SPARQL, there are five types of queries: Select, Ask, describe, construct, and update. The data source depends on the type of SDB used: table of triplets (single table or several tables if there is standardization) for a SDB of type I, a set of tables for SDB of types II and III. We can encapsulate this disparity using the SDB formalization proposed above. The results of a query are a set of tuples or a value.

In an abstract and formal way, we can define a query by a quadruplet *<E, DS, T, R>* where E is the algebraic expression of the query, DS is the data source (a *SDB <MI, I, Sch, Pop, MSMI, MSI, Ar>*), *T* is query type (select and ask) and *R* is set of tuples results from query execution. T is part of the expression for *E*. The set of queries Q in the materialized view problem is therefore a set of these quadruplets. In the resolution of the materialized view problem, q query rewriting can then be expressed by: *<E, W, T, R>* where *E'* is a new algebraic expression of the query q on the views, *W* is the set of optimal views on which the query will be executed, *T* is query type (select and ask) and *R* is set of tuple results of query execution. If a query is not completely covered by the views and must be covered by database tables, then *W* will therefore be composed of the optimal set of views and the tables of the database.

3.5. Classification of SDB queries

In the standard semantic database query language, SPARQL [23], [24], queries are expressed in the form of a series of triplets containing variables followed by filters, ordering clause, sorting or aggregations. The sequence of these triples is interpreted as a sequence of joins. The interpretation of these queries certainly depends on the representation of the SDB but these queries can be reduced to queries of the selection-projection-join form. We distinguish the following types of requests according to their interests.

a) Select type queries: these are queries composed of a single triplet pattern i.e., they are of the form *select ?x where (?x, !p, !o)* where *!c* means that *c* can be a constant or a variable). In semantic database of type I, it is a selection on the table of triplets, in semantic database of type II, it is also a selection if the property is set. The case where the property is not set returns the entire database and is not interesting. On the other hand, in semantic database of type III, these queries can be interpreted as a union of the selections made on the different tables relating to the class of the property *p*.
b) Joins type queries (AND): these are queries made up of at least two triplet patterns sharing at least the same variable. In all semantic databases, they are interpreted as natural joins. In semantic database of type III, they can also use union operations.
c) Left outer joins (OPTIONAL) type queries: these are selection or join queries containing at least one optional triplet pattern. In all semantic databases, they are interpreted as left outer joins. In semantic databases of type III, they can use union operations.
d) Filter type queries: these queries are characterized by the presence of the Filter keyword. The selection criterion expression concerns various types of data and operators (=, >=, etc.).
e) Negation type queries (FILTER, BOUND): these are queries marked by the presence of a pattern with Filter Not Exist (…) or the presence of the MINUS operator. Evaluating queries with a pattern with Filter Not Exist (…) involves finding a solution for the graph pattern and eliminating solutions that do not match the filter. For queries with MINUS, we remove from all the results of the first expression, the compatible results of the second expression.
f) Distinct type queries: these are queries whose duplication of tuples is not authorized in the results. They are recognizable by the keyword DISTINCT or UNIQUE.
g) Partial result type queries: these queries contain Limit or Offset keywords which delimit their results.
h) Sort type queries: these are queries whose results are sorted according to the order indicated by the ORDER BY clause.
i) Aggregate type queries: these are queries including the group by clause which announces the grouping of results on the indicated variable. There, we find the aggregate functions (COUNT, SUM, MIN, MAX, AVG, GROUP_CONCAT and SAMPLE). This type of query is taken into account from SPARQL 1.1 [25].

Of all these types of queries, join queries are by far the ones that consume the most resources (CPU, memory, and I/O).

4. CONCLUSION

Materialized views are a very important technique for the physical design and administration (tuning) of any database. We focused on the formalization of the materialized view selection problem in the context of ontology-based databases. We proposed a formalization of the problem and defined the different components. We also proposed a resolution schema to the problem. Our formalization is intended to be a generic framework that can circumscribe the different methods for solving the problem of selecting views to persist. A characterization of queries is discussed. This work on semantic databases can be applied to other types of databases. As perspectives of this work, it would be good to approach this formalization to distributed databases mainly cloud databases. NoSQL databases which use different schemas for data representation could also be a good field of prospecting works.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Bery Leouro Mbaiossoum	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓		✓	
Narkoy Batouma	✓	✓				✓	✓	✓	✓	✓	✓	✓		
Atteib Doutoum Mahamat	✓	✓	✓	✓		✓			✓	✓	✓			
Ouchar Cherif Ali	✓	✓	✓	✓						✓				
Lang Dionlar	✓	✓	✓	✓						✓				
Ladjel Bellatreche	✓				✓		✓			✓		✓		

C :	**C**onceptualization	I :	**I**nvestigation	Vi :	**Vi**sualization
M :	**M**ethodology	R :	**R**esources	Su :	**Su**pervision
So :	**So**ftware	D :	**D**ata Curation	P :	**P**roject administration
Va :	**Va**lidation	O :	Writing - **O**riginal Draft	Fu :	**Fu**nding acquisition
Fo :	**Fo**rmal analysis	E :	Writing - Review & **E**diting		

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.

REFERENCES

[1] E. Baralis, S. Paraboschi, and E. Teniente, “Materialized view selection in a multidimensional database,” *VLDB*, vol. 97, 1997.

[2] H. Gupta, “Selection of views to materialize in a data warehouse,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 1186, Springer Berlin Heidelberg, 1997, pp. 98–112.

[3] G. Wang *et al.*, “Temporal graph cube,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 12, pp. 13015–13030, Dec. 2023, doi: 10.1109/TKDE.2023.3270460.

[4] S. Bimonte, E. Gallinucci, P. Marcel, and S. Rizzi, “Logical design of multi-model data warehouses,” *Knowledge and Information Systems*, vol. 65, no. 3, pp. 1067–1103, Nov. 2023, doi: 10.1007/s10115-022-01788-0.

[5] P. Srinivasarao and A. R. Satish, “Multi-objective materialized view selection using flamingo search optimization algorithm,” *Software - Practice and Experience*, vol. 53, no. 4, pp. 988–1012, Dec. 2023, doi: 10.1002/spe.3174.

[6] A. Yadav and B. Singh, “Improve the performance of multidimensional data for OLAP by using an optimization approach,” in *AIP Conference Proceedings*, 2023, vol. 2745, no. 1, p. 20016, doi: 10.1063/5.0132474.

[7] R. Castillo and U. Leser, “Selecting materialized views for RDF data,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6385 LNCS, 2010, pp. 126–137.

[8] F. Goasdoué, K. Karanasos, J. Leblay, and I. Manolescu, “View selection in semantic web databases,” *Proceedings of the VLDB Endowment*, vol. 5, no. 2, pp. 97–108, Oct. 2011, doi: 10.14778/2078324.2078326.

[9] V. Dritsou, P. Constantopoulos, A. Deligiannakis, and Y. Kotidis, “Optimizing query shortcuts in RDF databases,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6643 LNCS, no. PART 2, Springer Berlin Heidelberg, 2011, pp. 77–92.

[10] McGuinness, D. L., and F. Van Harmelen, “OWL web ontology language overview,” *W3C recommendation*, vol. 10, no. 10, 2004.

[11] “Industrial automation systems and integration parts library Part 42 : description methodology : methodology for structuring parts families,” *ISO-13584-42*, 1998.s

[12] B. McBride, “Jena : Implementing the rdfmodel and syntax specification,” *In Proceedings of the 2nd International Workshop on the Semantic Web*, 2001.

[13] C. Murray, “Oracle spatial resource description framework (rdf),” *Oracle Corporation*, 2005.

[14] C. Murray, “Oracle database semantic technologies developer’s guide i,” *Oracle Cor¬poration*, 2008.

[15] J. Broekstra, A. Kampman, and F. Van Harmelen, “Sesame: A generic architecture for storing and querying RDF and RDF Schema,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 2342 LNCS, Springer Berlin Heidelberg, 2002, pp. 54–68.

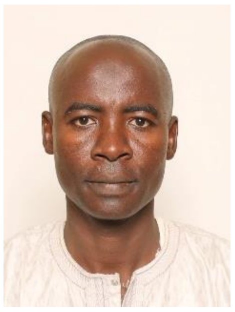
[16] S. Alexaki, V. Christophides, G. Karvounarakis, and D. Plexousakis, “The ics-FORTH RDFSuite: Managing voluminous RDF description bases,” *In Proceedings of the 2nd International Workshop on the Semantic Web*, pp. 1–13, 2001.

[17] J. Lu, L. Ma, L. Zhang, J.-S. Brunner, C. Wang, and Y. P. Y. Yu, "Sor: a practical system for ontology storage, reasoning and search," *Proceedings of the 33rd international conference on Very large data bases (VLDB'07)*, pp. 1402–1405, 2007.
[18] D. Theodoratos and T. Sellis, "Designing data warehouses," *Data and Knowledge Engineering*, vol. 31, no. 3, pp. 279–301, Nov. 1999, doi: 10.1016/S0169-023X(99)00029-4.
[19] B. Mbaiossoum, L. Bellatreche, S. Jean, and M. Baron, "Comparaison théorique et empirique de systèmes de bases de données sémantiques," *Ingenierie des Systemes d'Information*, vol. 18, no. 3, pp. 39–63, Jun. 2013, doi: 10.3166/ISI.18.3.39-63.
[20] G. Pierra, H. Dehainsala, Y. Ait Ameur, and L. Bellatreche, "Bases de données à base ontologique. Principe et mise en oeuvre," *Ingénierie des systèmes d'information*, vol. 10, no. 2, pp. 91–115, Apr. 2005, doi: 10.3166/isi.10.2.91-115.
[21] F. Frasincar, G. J. Houben, R. Vdovjak, and P. Barna, "RAL: An algebra for querying RDF," *World Wide Web*, vol. 7, no. 1, pp. 83–109, Mar. 2004, doi: 10.1023/B:WWWJ.0000015866.43076.06.
[22] M. L. Gillenson, *Fundamentals of database management systems*. John Wiley & Sons, 2023.
[23] E. Prud'hommeaux, *Sparql query language for rdf*. http ://www.w3.org/TR/rdf-sparql-query/, 2005.
[24] E. Prud and A. Seaborne, "SPARQL query language for RDF," 2006.
[25] S. H. Garlik, A. Seaborne, and E. Prud'hommeaux, "SPARQL 1.1 query language," *World Wide Web Consortium*, 2013.

BIOGRAPHIES OF AUTHORS

Bery Leouro Mbaiossoum is a Chadian teacher-researcher, specializing in the field of computer science and its applications. He has ascended the academic ranks with brilliance, earning a doctorate in computer science and applications from the École Nationale Supérieure de Mécanique et d'Aérotechnique (ENSMA) in Poitiers, France. His professional career is equally impressive. He has held various positions of responsibility at the University of N'Djaména, where he has been teaching and conducting research since 2005. Notably, he has served as the Head of the Computer Science Department and the Master's Program Director in Computer Science. His expertise spans a wide range of areas, including databases, computer systems and networks, design methods, ontology models, and programming languages. He is also an expert in ICT-mediated teaching, demonstrating his commitment to integrating modern technologies into education. He is proficient in several languages, including French and English, and has solid experience in network administration and computer system security. In addition to his teaching and research activities, he has also worked as a local service provider for CBN (Canadian BankNote) and as a CISCO instructor at the N'Djaména Academy. He has also completed several internships in research laboratories and companies, allowing him to gain valuable practical experience. He is the author of numerous scientific publications in international journals and conferences. His work focuses on topics such as ontology-based databases, the Semantic Web, and geographic information systems. He can be contacted at email: mbleouro@yahoo.fr.

Narkoy Batouma is a male lecturer and researcher in the Computer Science Department at the University of N'Djaména. His research primarily focuses on advanced computer systems and database management. Key areas of his work include the optimization of database systems (ranging from relational databases to big data), quality of service (QoS) management, and the behavioral adaptation of distributed applications in response to varying resource availability. His goal is to enhance the performance, reliability, and efficiency of complex distributed computing systems. He can be contacted at email: batoumanarkoy@yahoo.fr.



Atteib Doutoum Mahamat is a highly skilled computer science expert and accomplished academic, holding a Ph.D. in computer science from the University Blaise Pascal in Clermont-Ferrand, France. He is a CAMES Assistant Professor and currently serves as the Director of Judicial Statistics and Information Technology, while also working as a teacher-researcher at the Faculty of Exact and Applied Sciences (FSEA) at the University of N'Djaména, Chad. His expertise spans a wide range of areas, including research in remote sensing imagery, environmental mapping, the design of new change detection methods for satellite images, knowledge representation and management, optimization, graph theory, and language theory. He also possesses significant experience in computer science education, gained in both Chad and France. His professional career is marked by diverse experiences. He has served as the Head of the Computer Science Department at FSEA and has participated in numerous national and international events and workshops, particularly focusing on the digital transformation of justice and human rights. He has also undertaken postdoctoral scientific mobility in France. He has extensive experience in university teaching, where he has supervised numerous undergraduate and graduate students. He has also held visiting lecturer positions at several higher education institutions in Chad and France. He is the author of several scientific publications in international journals and has presented at numerous conferences. His research focuses on remote sensing, image processing, and artificial intelligence. He can be contacted at email: mabdoutoum@gmail.com.

Ouchar Chérif Ali Doctor of computer science, is a lecturer-researcher and an expert in proactive cybersecurity, threat intelligence, predictive cyberattack analysis, and the integration of AI in both healthcare and digital governance. With a background including a Bachelor's, Master's, and Doctorate in Computer Science (CAMES), he brings over ten years of professional experience in implementing innovative solutions across public and private sectors, higher education, and technology consulting, demonstrating flexibility, operational capability, and collaborative leadership. He can be contacted at email: alioucharcherif@yahoo.fr.

Lang Dionlar is a Chadian computer science researcher and educator, currently pursuing a doctorate in complex systems modeling and simulation at ESP, Dakar. He holds an engineering degree in computer science from the University of Khoms/Tripoli. His professional experience includes teaching positions at the University of N'Djaména, where he served as Head of the Computer Science Department. He has also worked in the private sector, including roles at SOGEA-SATOM and an IT assembly center in Libya. His possesses expertise in design methodologies (Merise, UML, EXPRESS), programming languages (Java, C++, Python), and database management (Oracle, MySQL). He is skilled in network administration (Windows, Linux), cybersecurity, and geographic information systems (GIS). His publications cover topics like ICT waste management, smart education, and service infrastructure for SMEs. He is trilingual, proficient in French, English, and Arabic, both spoken and written. He can be contacted at email: dionlar.lang@gmail.com.

Ladjel Bellatreche is an exceptional class full professor at the National Engineering School for Mechanics and Aerotechnics (ISAE-ENSMA), Poitiers - France, since September 2010. Before joining ISAE-ENSMA, he spent eight years as an Assistant and later an Associate Professor at Poitiers University, France. Since 2019, he has been also been a Part-time Professor at the Harbin Institute of Technology (HIT), China. From 2012 to 2022, he led the Data and Model Engineering Team at the Laboratory of Computer Science and Automatic Control for Systems (LIAS). Over the years, he has held several international positions, including visiting professor at the University of New South Wales, Sydney, and the Université du Québec en Outaouais. Additionally, he was a Visiting Researcher at Purdue University, USA, and the Hong Kong University of Science and Technology (HKUST). His research focuses on data science, artificial intelligence, knowledge graphs, recommender systems, query answering systems, and large language models. He emphasizes two critical non-functional properties: query performance and energy consumption, ensuring efficiency, scalability, and sustainability. He has authored over 350 publications in international conferences and journals, including IEEE TKDE, information systems, future generation computer systems, and data and knowledge engineering. Additionally, he serves as an Associate Editor for Data and Knowledge Engineering (Elsevier) and Knowledge and Information Systems (Springer). He has played key roles in major conferences such as ER 2025 (General Co-chair), IEEE Big Data, DASFAA, ADBIS, MEDES, FQAS, APWEB-WAIM, and DaWaK. He also served on the steering committees of ACM MEDES, ADBIS, DOLAP, BDA, and EDA. Notably, he is the founder of the International Conference on Model and Data Engineering (MEDI). Committed to fostering research in Africa and Asia, he co-supervise students from Algeria, Cameroon, Chad, Morocco, Tunisia, Togo, and Vietnam. He has secured research funding from the European Union, French National Research Agency, and industry partners. Since 2023, he has been the Program Director for the computer science for Avionics Master's program within the Master Aeronautics and Space at ISAE-ENSMA. He can be contacted at email: ladjel.bellatreche@ensma.fr.