On ontology based data integration: problems and solutions

IOP Conf. Series: Journal of Physics: Conf. Series **1203** (2019) 012059 doi:10.1088/1742-6596/1203/1/012059

On ontology based data integration: problems and solutions

A Gusenkov[1], N Bukharaev[2] and E Birialtsev[3]

[1] Associate Professor, PhD, Institute of Computational Mathematics and Information Technologies, Kazan Federal University, Kazan, Russia
[2] Associate Professor, PhD, Institute of Computational Mathematics and Information Technologies, Kazan Federal University, Kazan, Russia.
[3] R&D Director Gradient Ltd, PhD, Kazan, Russia

E-mail: [1]gusenkov.a.m@gmail.com, [2]boukharay@gmail.com, [3]igenbir@yandex.ru

Abstract. In the article essential problems of integrating heterogeneous data, arising in development of corporate databases intellectual access systems, are considered. In addition to the common structural problems, caused by variety of data organization, special attention is paid to the less obvious linguistic problems, caused by differences in data notation. A unified approach to overcoming such problems by sequential application of explicit definition of semantics, is described. This approach was tested in development of an intelligent search system for the TATNEFT oil-producing corporation; the system implementation showed high relevance of search results together with an adequate reactivity.

1. Introduction

The relational data model, based on the Codd's relational algebra and the normalization theory [1], is represented by a wide range of various relational database management systems (RDB MS). Large corporations simultaneously use today a significant number (up to several hundred) of various information systems. All of them, as a rule, have their own databases to store input and operational information. This raises the necessity to solve the problem of integration, i.e. need to ensure the possibility of unified user representation of heterogeneous data. Let's note the initial closeness of this problem to the problem of developing intellectual search systems, which provide end users access to information in a form, that does not require special knowledge and skills. Going forward we will consider these problems in a unity.

There are few approaches developed to solve problems of database integration. Among those the most well-known are the following.

Federated databases [2, 3] is an approach that involves implementation of the mutual relations between each of the databases and all the others. The main problem lies here in the laboriousness of writing numerous fragments of software code that ensure the translation of requests from one DBMS in terms of the other.

When choosing the *data warehouses* approach [2, 4-6], we store copies of parts of information from several databases in a single database (usually with preprocessing such as filtration, join, aggregation). During the copying process, the data arrays are transformed in order to align their structure with the general storage scheme.

Mediators [2, 7] are software components that support virtual database management. The mediator translates each user request into one or more requests addressed to different DBMSs. After that, the mediator synthesizes from the results of processing these queries the response to the original query.

The data warehousing and mediator technologies presume creation of software shells that perform the function of extracting information from primary sources (i.e. specific DBMSs). When implementing a data warehousing scheme, "built-in" queries are usually used to access sources and supply information to the data warehouse. In systems based on mediators, we meet more complex shells able to receive various requests sent by the mediator to the source. Here, parameterized query patterns as well as high-level specification shell generators are used. To construct a query plan (i.e., sequences of interaction between the mediator and the shells), various original strategies are used.

Note that all these approaches presume representation of the data integration problem solution in a procedural form. Every time when the structure of data sources or the composition of integral queries is changed, high qualified professional man labor is required to code the changes.

In accordance with the classification, proposed by M.R. Kogalovsky in [8], the solutions mentioned above should be attributed to the intermediate *logical level of data integration*. Unlike integration at the low *physical level*, which implies direct conversion of heterogeneous data into a single physical presentation format, this type of integration presumes implementation of procedures for accessing heterogeneous data in terms of some global logical scheme of their joint presentation.

Integration at the *semantic level* implies presence of an explicit formal description of the data semantic properties in terms of the subject domain ontology, separated from the access mechanism [9, 10]. This high-level data model forms the basis of the user interface with powerful intelligent search capability. In general, this approach, which underlies the Semantic Web, should be considered as a promising direction for building conceptual models of corporate information systems [11, 12].

In particular, being shared resources, such ontologies are best suited to serve as the base of common interface for heterogeneous data sources access, which determines their core place in solving data integration problems. Below we analyze in more detail a number of such problems, the successful solution of which we see in the consistent usage of explicit description of semantics.

As a preliminary general characteristic of the approach described below, let's note that to solve the problems of integration, along with information on the subject domain, we use auxiliary information resources such as a logical database model, physical database models and thesaurus of user terminology. All of them are presented in an unified ontology formalism – namely, OWL/DL language [13]. Let's emphasize, that in construction of these ontologies, in accordance with the specifics of the problems listed below, the computer linguistics apparatus is widely used along with traditional methods of software engineering.

2. Problems of RDBs integration

Traditionally, access to relational databases is made either through predefined forms (such as QBE [14]) or directly through the SQL query language. In order to succeed, one should understand the database structure and be able to generate syntactically correct and semantically meaningful queries in a RDBMS specific SQL dialect. On the other hand, it is also important to be able to interpret the business meaning of the results of query execution. In other words, this activity, requiring special skills and knowledge both in the field of professional programming and the specific business area, is unlikely could be fully automated.

As the most non-trivial and significant, we can distinguish here the problems of the following kind.

- *Problems, caused by variety of database structures.* The process of database normalization (stimulated more by technical reasons of computational efficiency, such as compact storage and access speed) can be implemented in various ways. The study of a specific version of partitioning of data by tables, needed to generate meaningful queries, requires considerable time and skill.
- *Problems, caused by variety of notations.* Even in the case of using the same natural language, the names of tables, columns and vocabulary elements in different databases may differ both from each other, as well as from the terms usually used by the subject domain professionals. The translation of multiple terms, required to formulate a syntactically correct query, turns out to be a very non-trivial task.

2.1. Structural problems of RDBs integration

Let's consider in more detail the first kind of problems on the example of some imaginary company "Center" and its subdivisions "East", "West", "North" and "South", which (for some historical reasons) use different databases. The company collects information on sales volumes from these RDBs into the target table (see table 1); the columns YEAR, MONTH and DIVISION form the primary key of the table.

Table 1. Aggregate balance.

YEAR	MONTH	DIVISION	EXPENSES	INCOME

The first problem encountered when transferring information between databases is the possible differences in the physical structure of the database tables. The attributes (columns or set of columns) can be distributed differently across tables, due to the flexibility of the database normalization rules [15, 16]. Different RDB designers can obtain various normalization schemes depending on their main business goals, experience and the specific requirements. So, the case, when a certain group of non-key attributes is stored in the main table or is placed in a separate table, is widespread in practice. For instance, our source table (see table 1) can be divided into two (see table 2a and 2b):

Table 2. Balance: (a) income; (b) expenses.

(a)

YEAR	MONTH	DIVISION	INCOME

(b)

YEAR	MONTH	DIVISION	EXPENSES

We have presumed an idealized picture, where initial sets of attributes were assumed to be the same and the RDBs involved in integration differed only in the structure of the tables. For real databases this condition is not always met, which imply at least two more problems.

The first one is related to the possible differences in the levels of abstraction when designing databases. The data structures of a type, having several slightly different subtypes, can be represented as separate tables for each subtype or a single table, in which the attributes of all subtypes are included. In the latter case, a column-selector, the value of which determines the subtype of a specific record, is included into the table.

In general, *segmentation by key*, i.e. the case when the table is divided into a set of tables, corresponding to the value of some selector column of the scalar type, is quite common in practice [15, 17]. As an example, consider again the table, containing information on the sales volume of the "Center" company, where the columns YEAR, MONTH and DIVISION form the primary key of the table. Usually, the value of the key column in one form or another is present in the table name. After applying segmentation by key, we may get the following set of tables (see table 3a and 3b):

Table 3. Balance, segmented by (a) year value; (b) division name.

(a)

Year	**Table columns**			
2013	MONTH	DIVISION	EXPENSES	INCOME
2014	MONTH	DIVISION	EXPENSES	INCOME
2015	MONTH	DIVISION	EXPENSES	INCOME

(b)

Division	**Table columns**			
"East"	YEAR	MONTH	EXPENSES	INCOME
"West"	YEAR	MONTH	EXPENSES	INCOME
"North"	YEAR	MONTH	EXPENSES	INCOME
"South"	YEAR	MONTH	EXPENSES	INCOME

Another frequently used possibility is transference of some key column value to the column name. In the example under consideration, this can be done in several ways. In particular, the income and expense tables can be presented in the following form (see table 4a and 4b):

Table 4. The names of (a) the divisions or (b) the months moved from the key to the column names.

(a)	YEAR	MONTH	EAST	NORTH	WEST	SOUTH
(b)	YEAR	DIVISION	JANUARY	FEBRUARY	…	DECEMBER

Obviously, combinations of the listed cases, when a combination of key parameters values forms the columns or tables names, are also possible.

One more particular problem is caused by non-atomicity of attributes. So, in our case, we considered the YEAR and MONTH attributes as separate, but other database developers could well consider them as a single attribute and present the same information in the following way (see table 5):

Table 5. Non-atomic attribute "Period".

PERIOD	DIVISION	EXPENSES	INCOME
January 2011			
February 2011			
etc…			

Here the PERIOD field contains two independent attributes, so to solve the integration problem it is necessary to perform the lexical analysis of the value. There are plenty examples of the kind. For example, surname, first name and patronymic can be represented as one, two or three attributes; the name of a company may include an indication of its form of ownership etc. There are even more complex cases such as representation of postal address, which can contain a lot of optional or rare elements.

Considering the problem of atomicity of attributes, let's also mention the adjacent problem of attribute scaling. Suppose that four divisions of our "Center" company are located in the US, Norway, Saudi Arabia and China. Then, quite naturally, incomes and expenses will be calculated in different monetary units. In addition, it is very likely that the time calculus will also differ. In addition, it is more natural for the United States to count weeks rather than months, and so on. In the case of a one-dimensional numeric scale, such problems can be solved simply by transition to some common measure. Much more complex problems of defining exact semantic arise in the case of qualitative notions, such as color, shape, taste, etc.

And, the last not least, let’s also note, that some valuable information may not be stored in the databases at all, but assumed to be understood by default. So, if the subdivisions of our company initially recorded income and expenses independently, then their databases store no information about the division name, which otherwise would have constant value for all records. Thus, when integrating the four databases of the divisions, we could deal with information of a completely similar structure (see table 6), but related to different units.

Table 6. Company division name understood by default.

YEAR	MONTH	EXPENSES	INCOME

Here we consider the need of unambiguous definition of semantics mainly in the context of data integration problems. But obviously it has much more general meaning; different assumptions (say, in measurement) can easily lead us to a disaster.

As we have already mentioned above, a lot of data integration problems is caused by variety of database objects naming. It is difficult to expect that independent developers will name in exactly the same way objects of the same meaning (such as tables, columns and domains). Vice versa, in such cases it usual to use various synonyms, abbreviations and grammatical constructions. So, in the example under consideration, the source table could easily have one of these forms (see table 7):

Table 7. Use of synonyms in the database objects names.

YEAR	MONTH	UNIT	EXPENDITURE	INCOME
or				
YEAR	MONTH	DIVISION	COSTS	RETURN

The solution of such problems of lexical "multilingualism" is far from a trivial one. It leads us into the sphere of computer linguistics and clearly deserve separate consideration.

2.2. Linguistic problems of RDBs integration

Before turning to description of specific problems of linguistic nature, let's note first two key issues helping us to resolve the problems of "multilingualism" (i.e. the variety of meaning representation forms), mentioned above:

- problems of this kind can be formulated in terms of *linguistic relations* (such as synonymy, antonymy etc);
- methods of computer linguistics can be used to extract such relationships to extracted from texts in natural language *semi-automatically*.

Further consideration of the linguistic problems will be carried out using the example of real databases of the TATNEFT oil and gas producing corporation. In particular, the problem of transferring data from the MS SQL database to two Oracle databases was investigated. The Oracle databases taken together stored information, approximately equivalent to those in the MS SQL database. Some parameters of their structure are given in table 8.

Table 8. Composition of the studied databases.

	MS SQL DB	Oracle DB1	Oracle DB2
Number of tables	644	219	95
Number of columns	11688	2028	524

A typical description of the structure of the tables under consideration is given in table 9.

Table 9. Typical table description.

Column ID	**Column descriptor**
NC	Well no
GOD	Year
MES	Month
PL	Reservoir code
SPEX	Method of operation
DN	Oil mining

DW	Water mining
DG	Gas mining
KDEX	Work hours
PLB	Tailings density

It is obvious that the columns identifiers here are not informative for the task of identifying their semantics. Much more informative are the attribute natural language descriptions, stored in the field comments. Along with literary language vocabulary, such descriptions usually use the professional terms from the subject domain. Analyzing the attribute descriptions is the only possible way to automatically identify their semantics. Thus, the data integration problem requires consideration from a linguistic point of view and application of methods of computer linguistics to get its adequate solution.

Even a superficial analysis of table 8 shows that the approaches to database development in MS SQL and Oracle DBMS significantly vary. Note that the numbers of columns and tables in the databases differ several times. Meanwhile, comparison of the numbers of terms, contained in the columns definitions and vocabulary elements given in table 10, shows that the numbers of semantically different terms in these databases are almost equal (24035 and 24504).

Table 10. Total number of terms in the databases.

Records	**Quantity**
Total amount	48 629
Those from the Oracle RDBs	24 035
Those from the MS SQL RDB	24 504

Direct coincidence of terms definitions was observed in less than one percent of cases. At the same time, the analysis carried out at the level of lexemes showed that the concurrence of the used lexicons is much more significant (see table 11). The parsing to tokens with elimination of morphological differences showed a coincidence of vocabulary in about 40% of cases.

Table 11. Coincidence of vocabularies.

Elements	**Quantity**
Words in total	163 431
Unique roots	6 078
Common in the Oracle and MS SQL RDBs	2 518
In Oracle RDB only	2 423
In MS SQL RDB only	1 137

It's interesting to understand the nature of the remaining differences in the lexical composition. Let's consider the reasons for them on the example of the mapping between the MS SQL database table and the Oracle database tables terms, performed expertly. The table 12 demonstrates many lexical-semantic relationships between table column definitions of the databases. Note that the simplest for analysis and presentation synonymy relation is quite rare. For instance, in the above example, the pair of terms "Operating mode" and "Development method", considered in the professional context, are synonyms.

Table 12. Example of terms comparison.

<Table 1> column names	**Lexical and semantic relation**	**<Table 2> column names**
Well no. Reservoir code	Meronymy	Development object
Year Month	Hyponymy	Operation period
Method of exploitation	Synonymy	Method of development
Water extraction		Fluid type
Oil extraction	Conversion, hyponymy	Fluid return
Gas extraction		
Work hours	Antonymy	Percentage of downtime
Density of tailings		

Other matches demonstrate much more complex relationships. Say, to match the "Year" and "Month" columns in <Table 1> and the "Operation Period" column in <Table 2>, one must take into account, that the Year and Month are hyponyms for the Period, which refer specifically to the period of oil production, but not to another event or interval (such as, for example, putting the well into operation).

To correlate the attributes "Water production", "Oil production", "Gas production" from <Table 1> and "Fluid type" and "Fluid return" from <Table 2>, it is necessary to understand that in this professional context "water", "oil" and "gas" are hyponyms of the term "fluid", and that "extraction" and "return" are conversions, expressing the point of view of the process: "The well extracts oil from the deposit", "The oil field gives oil to the well".

The correspondence of the columns "Hours of operation" and "Percentage of downtime" is based on the antonymy of the terms "Work" and "Downtime". Knowing the downtime of the well and the month to which it relates to, one can calculate the well operating time. An additional complication caused by the fact, that quantitative measurements of the operation and downtime of the well are expressed in different time units – hours in absolute units, and percentage in relative units.

The correspondence between the pair of "Well Number" and "Reservoir Code" and the "Development Object" terms reflects the fact that in the professional slang the development object term means production of fluid from a particular formation at a specific well. Thus, the "Development Object" includes the formation and the well as its constituent parts. Thus, in this case, there is a relation of meronymy between the terms. In general, relations of meronymy between the columns definitions of the types "Object-Role", "Process-Result", etc. were met quite often in the investigated databases.

3. On fundamentals of the semantic approach

Let's make some intermediate conclusions. Problems of RBD integration like the ones we've met above are not eliminable. They should be resolved either individually or, preferably, more or less universally and uniformly, which presumes a higher level of abstraction.

In both cases information required to resolve the relevant collisions must be presented in any data integration system. In the traditional approach representation of that knowledge is incorporated into the program code. It is specific and implicit. In the systematic application of the semantic approach it is holistic, explicitly formalized and separated from other components of the system; especially, from the universal mechanism for providing data access. In other words, the corresponding information presented in the ontology formalism is considered here as a configurable system parameter. This

makes the logic of system development more transparent. The system itself becomes much more stable with respect to inevitable business changes.

Note that these arguments become even more valuable when we turn to development of intellectual search information systems. In any case intellectual search presumes semantic search, i.e. possibility of the end users to communicate with an information system on the language most suitable to achieve main goals of the system usage.

Let's demonstrate the declared statements on a real-life example of developing such system for a large oil company.

3.1. System architecture

The main components of the system are:

- *the linguistic thesaurus* of the subject domain, formally defining the language of the user system communication;
- *the subject domain ontology*, defining semantics of professional terminology;
- *the universal RDB ontology*, describing the basic concepts of relational databases in the ontology formalism; each of RDBs to be integrated considered as an instance of the universal ontology;
- *the algorithm of intellectual access to the set of RDBs*, generating SQL-queries for given end user queries on the professional natural language dialect.

Let's describe briefly the role of these system components in solving the problems we described earlier. More detailed description of functionality and architecture of this system can be found in [18, 19, 20].

3.2. Linguistic thesaurus

The linguistic thesaurus gives formal definition of the subject domain language, serving as the basis of the end user interface.

The linguistic thesaurus was created according to the principles of Word Net thesauruses constructing [21]. The vocabulary of the subject domain, currently outnumbering 8,000 combined concepts, was built by combining the word forms of the defining industry standards Epicenter model of Petrotechnical Open Software Corporation [22, 23], and the TATNEFT specific word-forms, automatically extracted from the attribute descriptions from table-directories of the corporation RBDs.

For each word-form an input synonymic series (or *synset*) was defined to reflect variety of such descriptions in real RDBs, containing short phrases, abbreviations, technical abbreviations, etc. The thesaurus also includes the following linguistic relations of the word forms: hyponymy, part – whole, incompatibility, antonymy, convertibility, and homonymy (see details in [17, 23]). Such content is intended to provide information needed to unambiguously find semantics of end-user queries, expressed in a table form (see table 13).

Each line of such table defines a conjunctive member of the query predicate, expressing a certain restriction on the value, represented by the thesaurus word form.

Table 13. Query example.

Professional term	**Condition**
Well No.	10*
Date of commissioning	> 1.06.2015
Date of overhaul	
Expected oil production rate	>0
Actual oil production rate	

Such end user's requests can be considered as expressed in the professional dialect of natural language. Indeed, by using thesaurus word forms they refer exclusively to the semantics of the subject domain. They do not contain any references to the location or structure of the requested information stored in the databases.

3.3. Subject domain ontology

This ontology defines the formal semantics of the concepts of the subject domain – in our case, oil production.

As an initial ontology prototype, the Epicentre model of Petrotechnical Open Software Corporation (POSC) [24] was used. Recognized as the industry standard, it defines more than 1000 terms, related to the field of oil exploration and production. Based on the object-oriented concept of inheritance, the Epicenter model is represented by a set of ER-diagrams and text files in the EXPRESS language.

To automatically convert this model into the OWL ontology description language, a formal LR(1) grammar of the Epicenter model language was built and general conversion scheme has been developed [18, 25].

3.4. Universal RDB ontology and intellectual search algorithm

This ontology is used to separate the complexity of specific RDB structures from the universal mechanisms for their processing. In particular, it plays an important role in the design of the intellectual search algorithm.

This ontology describes the general theoretical notions of the relational database theory, namely

- concepts of TABLE, COLUMN, KEY, DOMAIN, corresponding to the main database objects, and
- basic relations between them:

  TABLE contains COLUMN;
  TABLE has a primary KEY;
  TABLE has a foreign KEY;
  (composite) KEY contains COLUMN;
  COLUMN has DOMAIN type.

Specific RDBS are treated as instances of the universal ontology; respectively, their tables, columns, keys, and domains are considered to be instances of basic concepts of the corresponding type.

Definition of the universal ontology also includes the following interpretation functions:

FI1: If TABLE1 has a primary KEY1 and TABLE2 has a foreign KEY1, then there is a TABLE3, containing all columns from TABLE1 and TABLE2.
FI2: If TABLE1 contains COLUMN1, then there is TABLE2 containing all the columns of TABLE1, except COLUMN1.

The first interpretation function implicitly defines the tables join operation, and the second function defines the relation projection operation. The latter is necessary to reduce the number of columns produced by joining tables to the desired one.

As we've seen earlier, an end user request refers to semantics of the subject domain, but it contains any no reference to the actual definition of data storage. Thus, the main task in implementation of intelligent search is location of data in the form of some set of the RDBs columns by description of its semantics.

In terms of the universal ontology, this problem can be expressed as the problem of finding such a sequence of applications of the FI1 and FI2 functions, which gives the result in the desired set of columns. This task can be easily reduced to the well-known class of oriented graph wandering problems. In our case, the graph vertices are instances of the TABLE concept, and the arcs denote

presence of a common key; the arcs are oriented according to “has primary key” relationship. Methods of solution of such problems are well-known; however, exponential growth of complexity of such methods should also been taken into account [26].

4. Conclusion

To solve the important problem of intellectual access to heterogeneous information resources, presented in the form of relational databases (RDB), several technologies have been proposed and industrial data integration platforms successfully implemented. Among the most well-known examples let's mention IBM Websphere Integration Server [6] and SAP Net Viewer [27].

Most of these approaches are based on direct procedural presentation of RDBs definition. Integration procedures (in the form of SQL scripts, java classes or other language components) explicitly refer here to the names of RDBs objects (such as tables, relations, columns, etc.). A significant drawback of such incorporation of the data definition into code is the need to manually modify the code when modifying the RDBs definition. Obviously, such lack of universality and uniformity of knowledge representation greatly complicates system maintenance and increases overall cost of integration projects.

The approach proposed in this article is based on explicit division of data integration knowledge into specific and universal. Information of the first type is represented by the once-tuned information system components, describing in the ontology formalism the subject domain and definitions of RDBs to be integrated. Universal information is represented by permanent procedural component, independent of any variable information. This simplifies the logic of system development and makes its maintenance more stable relative to changes in specific data. Such separation becomes even more critical in intellectual search systems development, where explicit definition of semantics of data is becomes the matter of the first importance.

To make this general ideology work well in the case of RDBs integration we analyze significant amount of specific problems, which system developers meet in the "real world" practice. We divide such problems into two parts, according to the methodology suitable to solve them. The problems of the first kind are caused by different ways of structuring information in RDBs. The problems of the second kind are caused by different ways of notation; they have essentially linguistic nature and computer linguistic methods are needed to deal with them.

The presented approach was tested during development of the intellectual search information system, implementing data integration for TATNEFT oil company RDBs. The system generates SQL queries formulated by the end users exclusively in terms of the subject domain [18], which puts inevitable theoretical restrictions on overall computational effectiveness. However, the system showed high relevance of the search results and good reactivity of the system in the case of searching information stored in 6-10 columns from different DDBs, which is quite enough for operational reference queries.

This suggests that the described approach can significantly reduce the time and laboriousness of corporate data integrating solutions. The immediate prospects for the method development are seen in its application to integration of semi structured data sources, such as object oriented databases and XML.

Acknowledgments

This work was funded by the subsidy allocated to Kazan Federal University for the state assignment in the sphere of scientific activities, grant agreement 1.2368.2018 and subsidy of the Russian fund of fundamental research, grant agreement 18-07-00964.2018.

References

[1] Codd E F 1970 Relation model of data for large shared data banks *Comm. ACM* **13** 6 pp 377–83
[2] Garsia-Molina G, Ulman J and Uidom J 2001 *Database Systems, The Complete Book* (Pearson Education)

[3] Sheth A P and Larson J A 1990 Federated databases for managing distributed, heterogeneous, and autonomous databases *Computing Surveys* **22** 3 pp 183–236
[4] Arhipenkov S Y, Golubev D and Maksimenko O 2002 *Data Warehouses. From Concept to Implementation* (Moscow: Dialog-MIFI)
[5] Sperley E 1999 Enterprise Data Warehouse: Planning, Building, and Implementation **1** (Prentice Hall PTR)
[6] WebSphere Business Integration Server Foundation URL: http://www-01.ibm.com/software/integration/wbisf/features/
[7] Wiederhold G 1992 Mediators in the architecture of future information systems *IEEE Computer* **25** 1 pp 38–49.
[8] Kogalovskij M 2010 *Methods of data integration in information systems* (Moscow: Russian Academy of Science Institute of Market Problems) URL: http://www.ipr-ras.ru/articles/kogalov10-05.pdf
[9] Gavrilova T and Horoshevskij V 2001 *Knowledge base of intelligent systems* (St. Petersburg: Piter Publishing House) 384 p
[10] Kogalovsky M R 2012 Ontology-based data access systems *Programming and Computer Software* **38** 4 pp 167–82 DOI 10.1134/S0361768812040032 URL: https://link.springer.com/article/10.1134/S0361768812040032
[11] Doan A, Madhavan J, Domingos P and Halevy A Ontology matching: a machine learning approach 2004 *Handbook on Ontologies in Information Systems* ed Staab S and Studer R (Springer-Verlag) pp. 397–16
[12] Doerr M, Hunter J and Lagoze C 2003 Towards a core ontology for information integration *J of Digital Information* **4** 1 pp 169–72
[13] W3C, OWL Web Ontology Language URL: https://www.w3.org/TR/2004/REC-owl-features-20040210/
[14] Buneman P 1997 Semistructured data *Proc of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symp on Principles of Database Systems* (Tucson Arizona United States May 11–15) pp 117–21
[15] Date C J 2004 *An Introduction to Database Systems* (Pearson Addison Wesley)
[16] Kogalovsky M R 2005 *Encyclopedia of Database Technologies* (Moscow: Finansy i statistika)
[17] Birialtsev E and Gusenkov A 2007 Integration of databases based on ontologies. *Scholarship Notes of the Kazan University. Physics and mathematics series* **149** (Kazan: Kazan University Press) pp 13–25
[18] Birialtsev E, Bukharaev N and Gusenkov A 2017 intelligent search in big data *J of Physics: Conf. Series* **913** 1 URL: http://iopscience.iop.org/article/10.1088/1742-6596/913/1/012010/pdf
[19] Gusenkov A M 2016 Intelligent search for complex objects in big data arrays *Electronic Libraries* **19** 1 pp 3–39
[20] Gusenkov A, Birialtsev E and Zhibrik O 2015 *Intellectual search in structured information arrays* (LAP LAMBERT Academic Publishing ISBN 978-3-659-76919-1) 129 p.
[21] Fellbaum C (ed.) 1998 *WordNet: An Electronic Lexical Database* (Cambridge: MIT Press) 423 p.
[22] Birialtsev E, Gusenkov A and Mironov S 2009 One approach to implementing unregulated access to relational databases *Proc. of Kazan School on Computer and Cognitive Linguistics TEL-2008* (Kazan: Kazan University Press) pp 10–23
[23] Birialtsev E and Gusenkov A 2007 Relational database ontologies. Linguistic aspect *Proc. of Dialog'2007 Conf.* (Moscow: RGGU Press) pp 50–53
[24] Energistics, Epicentre v3.0 URL: http://www.energistics.org/energistics-standards-directory/epicentre-archive
[25] Birialtsev E and Gusenkov A 2007 The construction of subject domain ontology based on the logical database model *Proc. of Knowledge-Ontology-Theory (ZONT-07) Conf.*

(Novosibirsk: Sobolev Mathematical Institute Press 1) pp 176–183
[26] Anderson J 2003 *Discrete Mathematics with Combinatorics* (Moscow: Williams Publishing House) 960 p
[27] SAP Net Viewer URL: https://www.tutorialspoint.com/sap/sap_net_weaver.htm