STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 2, 2009

# CONCEPTUAL KNOWLEDGE PROCESSING FOR DATABASES. AN OVERVIEW

CHRISTIAN SĂCĂREA AND VIORICA VARGA

Abstract. We present an overview of methods of Conceptual Knowledge Processing and their applications for databases, pointing out recent developments and joint research work in this field. Based on the mathematical theory of Formal Concept Analysis, we show how Conceptual Graphs can be used as a representation tool for both database structure and queries. Also we apply methods of Formal Concept Analysis to mine functional dependencies in relational databases. These methods are then discussed on several examples, presenting two software products developed so far, FCAFuncDepMine and CGDBInterface.

## 1. Conceptual Knowledge Processing

**1.1. Methods of Conceptual Knowledge Processing.** Formal Concept Analysis started in the early '80s as an attempt to restructure the classical lattice theory. The mathematical part of this theory quickly developed but the power of its expressiveness became clear by dealing with concrete problems of data analysis. Since then, Formal Concept Analysis has been extended to a powerful general framework for Conceptual Knowledge Processing and Representation.

According to [21], Conceptual Knowledge Processing is considered to be an applied discipline dealing with ambitious knowledge which is constituted by conscious reflection, discursive argumentation and human communication on the basis of cultural background, social conventions and personal experiences. Its main aim is to develop and maintain methods and instruments for processing information and knowledge which support rational thought, judgment and action of human beings and therewith promote the critical discourse. The word Conceptual in the name Conceptual Knowledge Processing underlines

Received by the editors: October 30, 2009.

2010 *Mathematics Subject Classification.* 68P15, 03G10.

1998 *CR Categories and Descriptors.* H.2.1 [**Database Management**]: Logical design – *Normal forms*; F.4.m [**Mathematical Logic and Formal Languages**]: Miscellaneous.

*Key words and phrases.* Conceptual Knowledge Processing, Formal Concept Analysis, Database Theory, Conceptual Graphs.

the constitutive role of the thinking, arguing and communication of human being in order to collect process knowledge. Conceptual Knowledge Processing is grounded on the mathematization of traditional philosophical logic, as a doctrine of concept, judgment and conclusion. The mathematical basis is build on Formal Concept Analysis, a mathematical theory of concepts and concept hierarchies, developed in the last 25 years, which has been proved useful in a large number of applications. Conceptual Knowledge Representation, Conceptual Classification, Analysis of Concept Hierarchies, Conceptual Identification, Conceptual Knowledge Inference, Acquisition and Retrieval are just some methods which can prove their usefulness.

Conceptual Knowledge Processing differs from other data analysis methods in that the emphasis is on recognizing structural similarities ([3]), turning a collection of data into a set of knowledge units called *concepts* and unfolding the subsequent encoded knowledge into a *conceptual hierarchy.*

Throughout this paper, *information* in the scope of Conceptual Knowledge Processing should be understood in the same way as in Devlin's book *Infosense - Turning Information into Knowledge* ([6]). Here, he briefly summarize this understanding in the formulas:

| | | |
|---|---|---|
| Data | = | signs + syntax |
| Information | = | Data + Semantics |
| Knowledge | = | Internalized Information + Possibility to use this information in order to acquire new knowledge. |

By this understanding, it becomes clear that Formal Concept Analysis is able to support the representation of information and knowledge ([20]). Moreover, it is important to point out the difference between information and knowledge, difference which constitutes the very essence of knowledge processing vs. data analysis. By this understanding, information is always *contextual*, while knowledge is always *conceptual*.

The mathematical theory of Formal Concept Analysis is based on a set theoretical semantics. Formal Concept Analysis starts with a formalization of contexts. Thus, a formal context is a triple $\mathbb{K} := (G, M, I)$. The set $G$ is called the set of objects, $M$ is the set of attributes, and $I$ is a binary relation between $G$ and $M$ indicating which object has which attribute, called incidence relation. A formal context should be understood as a formalization of a part of reality, containing the entire intended information about collected data.

Since a formal context is just a set of data and related information, we need to unfold a so-called knowledge map, by highlighting the basic knowledge units which are formal concepts. A formal concept is a pair $(A, B)$, consisting of two subsets, $A$ the extension, a subset of $G$, and $B$ the intension, a subset of

$M$. The formal extension $A$ contains all objects having the formal attributes collected in the set $B$. The formal intension contains all attributes valid for all formal objects collected in the set $A$. Concepts can be ordered in a hierarchy by the *subconcept - superconcept* ordering relation. A concept $(A, B)$ is called a subconcept of $(C, D)$, and $(C, D)$ a superconcept of $(A, B)$, if $A$ is a subset of $C$ or, equivalent, $D$ is a subset of $B$. We write $(A, B) \leq (C, D)$. A subconcept can be understood as a specialization of the superconcept, while the superconcept is the generalization of its subconcepts. The set of all formal concepts of a given formal context $\mathbb{K}$ is denoted by $\mathcal{B}(\mathbb{K})$. It is called *conceptual hierarchy* or *concept lattice*. Indeed, $\mathcal{B}(\mathbb{K})$ is a complete lattice ordered by the subconcept-superconcept relation. Conceptual hierarchies can be understood as a knowledge map for the information encoded in the formal context. Representing the conceptual hierarchy as a treelike structure enables navigation and activates background knowledge. It makes possible to unfold the conceptual structure of the entire data set in a very precise and coherent way. Conceptual hierarchies are valuable visualization methods for complex knowledge structures, enabling navigation from one knowledge object, i.e, a concept, to another concept. Conceptual hierarchies are also comprehensives knowledge maps, in which navigation is made along lines, from one node to another. Every time a node is hit, the corresponding information is revealed, i.e, what are the objects related to that node, and what are their attributes. By restricting and/or enlarging the set of attibutes or objects navigation becomes dynamic.

Implications in Formal Concept Analysis are an important feature for understanding the internal structure of data. The logic of data is part of the subsequent knowledge which is unfolded in the conceptual hierarchy. Nevertheless, it might be of interest to investigate dependencies between attributes in terms of implications. If $(G, M, I)$ is a formal context, let $A$ and $B$ be subsets of $M$. Formally, an implication is just a pair $(A, B)$ of subsets of $M$. We write $A \to B$. We say that the implication $A \to B$ holds in $(G, M, I)$ if and only if every object which has all the attributes from $A$ also has the attributes from $B$. It can be proven that the implications determine the conceptual hierarchy up to isomorphism and therefore offer an additional interpretation of the conceptual structure.

1.2. **Contextual Logic.** The interplay of the theory of Conceptual Graphs (CG) and Formal Concept Analysis (FCA) proved to be very fruitful in order to formalize the Elementary Logic, defined by I. Kant as "the theory of the three main essential functions of thinking: concepts, judgements and conclusions".

While FCA provides the mathematization of the classical theory of concepts, regarded as units of thoughts, CG are representing a formalization of the theory of judgements and conclusions.

Conceptual graphs can be understood as formal judgements. A conceptual graph is a labeled graph that represents the literal meaning of a sentence. Conceptual graphs express 'meaning in a form that is logically precise, humanly readable, and computationally tractable' ([12]). They serve as an intermediate language for translating computer-oriented formalisms to and from natural languages. With their graphic representation, they serve as a readable, but formal design and specification language.

In particular, they are graphs that consist of concept nodes, which bear references as well as types of the references. The concept boxes are connected by edges, which are used to express different relationships between the referents of the attached concept boxes. Sowa provides rules for formal deduction procedures on conceptual graphs; hence the system of conceptual graphs offers a formalization of conclusions too.

Hence, a conceptual graph is a bipartite graph having to kind of labeled nodes (concepts and relations) having the full description power of first order logic, FCA is a mathematical theory of deriving a conceptual hierarchy from a data table called *formal context.* Later on, these two theories have grown together, FCA being now part of a successful formalization of CG theory.

As Formal Concept Analysis provides a formalization of concepts, and as conceptual graphs offer a formalization of judgements and conclusions, a convincing idea is to combine these approaches to gain a unified formal theory for concepts, judgements and conclusions, i.e., a formal theory of elementary logic. In [19], Wille marked the starting point for a such a theory. There he provided a mathematization of conceptual graphs where the types of conceptual graphs are interpreted by formal concepts of a so-called power context family. The resulting graphs are called concept graphs. They form the mathematical basis for contextual logic.

Interaction with computers becomes more and more important in our daily lifes. The goal of conceptual graphs is to provide a graphical representation for logic which is able to support human reasoning. Possible applications of such a logical representation system have been described in [13], [14]. One interesting application is a consistent, graphical interface for database interaction, which has not been completely developed until today.

## 2. Conceptual graphs as database interface

The basic idea of using conceptual graphs as query interface to relational databases has been stated in the mid '70s by Sowa [11]. Almost twenty years later, after a consistent development of database theory, a first attemp to use conceptual graphs for relational databases was stated ([8]), namely it presents the scheme of a relational table as a conceptual graph and queries also as conceptual graphs.

The mathematization of conceptual graphs started by Wille, and continued by Prediger was completed by F. Dau [4]. Here, he studies a calculus for the mathematization of conceptual graphs and their equivalence to first order logic. In database theory (see [1]) the equivalence of the relational calculus and the relational algebra is well known. Modern database languages extend this expressivity, considering especially aggregate functions. F. Dau and J. C. Hereth [5] developed Nested Concept Graphs with Cuts to express aggregate functions and negation.

In our research, we made use of the previous results mentioned above, but we decided that there is need for specialization and slightly modification of the theory developed so far. Hence, we have modeled a new form of conceptual graphs. They enable vizualisation of both the structure of a relational database, and queries, allowing a user friendly representation of queries and structure of the relational database. A software product has been developed, presented at [18] named CGDBInterface, which offers a graphical tool to query an existing relational database. The aim of our software tool is to connect to an existing database by giving the type and the name of the database, a login name and password, then the software offers the structure of the database in form of a conceptual graph. The relation between tables was not represented with conceptual graphs in earlier works. We also have been successful in our attempt to represent the relation between tables too by a conceptual graph, method which has been implemented in the above mentioned software. We illustrate our results by the next examples. The mathematical background of the proposed model for database scheme and queries is under development, since the particular structure of the conceptual graphs we use, imposes some modifications of the theory stated in [4].

**Example 1.** Let be the next relational `Sales` database scheme. Figure 1 shows the conceptual graph obtained by CGDBInterface software.

```
Customers(CustomerID, CompanyName, ContactName, Address, Phone)
Suppliers(SupplierID, CompanyName, ContactName, Address, Phone)
Orders(OrderID, CustomerID, OrderDate, RequiredDate)
Products(ProductID, ProductName, SupplierID, UnitPrice,
                  UnitsInStock, UnitsOnOrder)
OrderDetails(OrderID, ProductID, UnitPrice, Quantity)
```

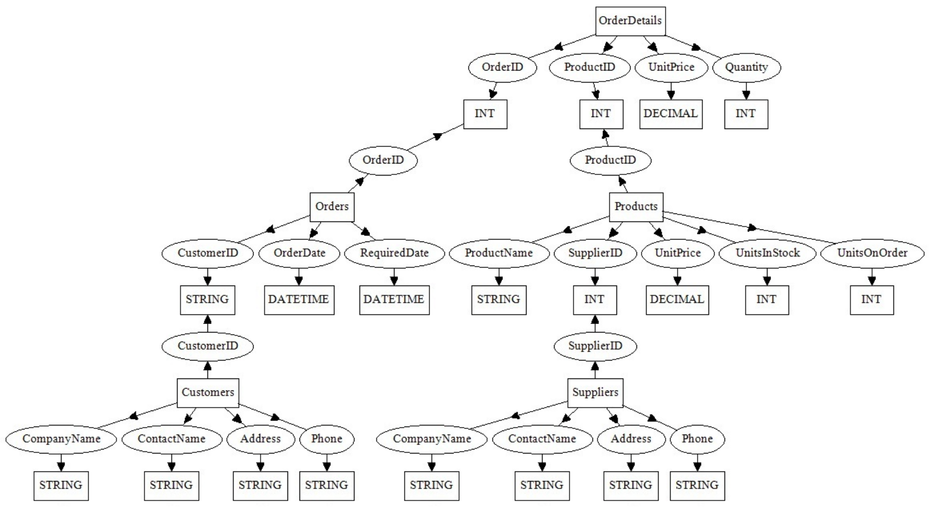
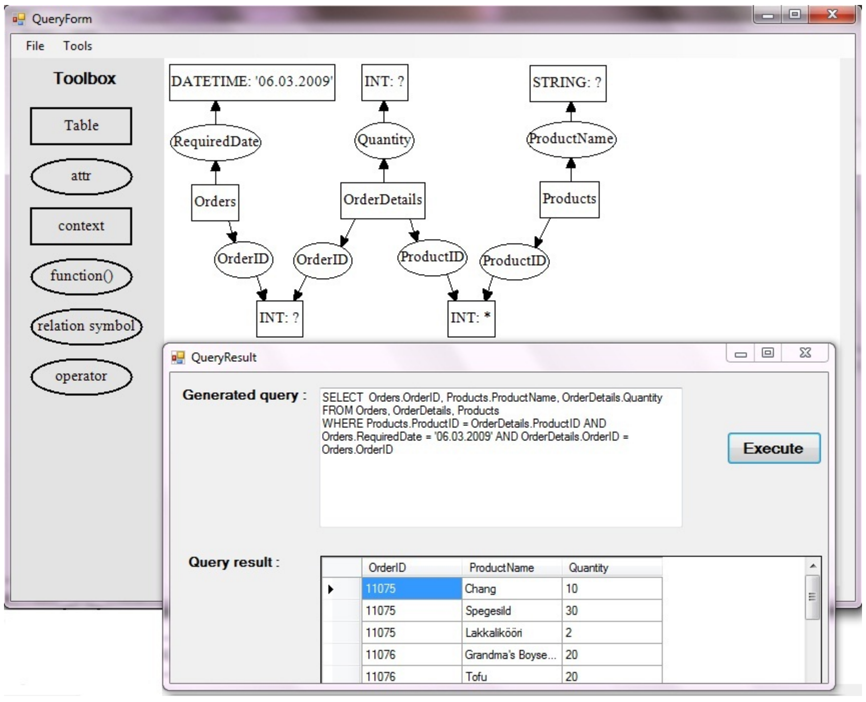


FIGURE 1. Conceptual graph for **Sales scheme**

Queries generated by software CGDBInterface covers all types of queries, starting with simple queries to very complex ones. The software uses hypostatic abstraction to model aggregation and nested queries.

**Example 2.** The next query is constructed by CGDBInterface, which has a graph editor. The SELECT SQL statement is generated by the software, then the generated query can be executed. In Figure 2 is presented the query, which searches for product names having been ordered for 3th of June, 2009. It gives the OrderID and order quantity too for every product. The selected attributes are marked by '?' in the query conceptual graph. To construct the query we have to join three tables: `Orders, OrderDetails` and `Products. The join attributes connect the tables.`

**Example 3.** The query in Figure 3 selects for every customer the number of products ordered by him. In order to use relations as objects in other relations, we have to reconsider these relations and to make use of a method introduced by Peirce, called *hypostatic abstractions*. *Hypostatic abstraction*, also known as hypostasis or subjectal abstraction, is a formal operation that takes an element of information, such as might be expressed in a proposition of the form $X$ is $Y$, and conceives its information to consist in the relation between a subject and another subject, such as expressed in a proposition of the form $X$ has $Y$-ness. The existence of the latter subject, here $Y$-ness, consists solely in the truth of those propositions that have the corresponding



```
SELECT  Orders.OrderID, Products.ProductName, OrderDetails.Quantity
FROM Orders, OrderDetails, Products
WHERE Products.ProductID = OrderDetails.ProductID AND
Orders.RequiredDate = '06.03.2009' AND OrderDetails.OrderID =
Orders.OrderID
```

| OrderID | ProductName | Quantity |
|---|---|---|
| 11075 | Chang | 10 |
| 11075 | Spegesild | 30 |
| 11075 | Lakkalikööri | 2 |
| 11076 | Grandma's Boyse... | 20 |
| 11076 | Tofu | 20 |

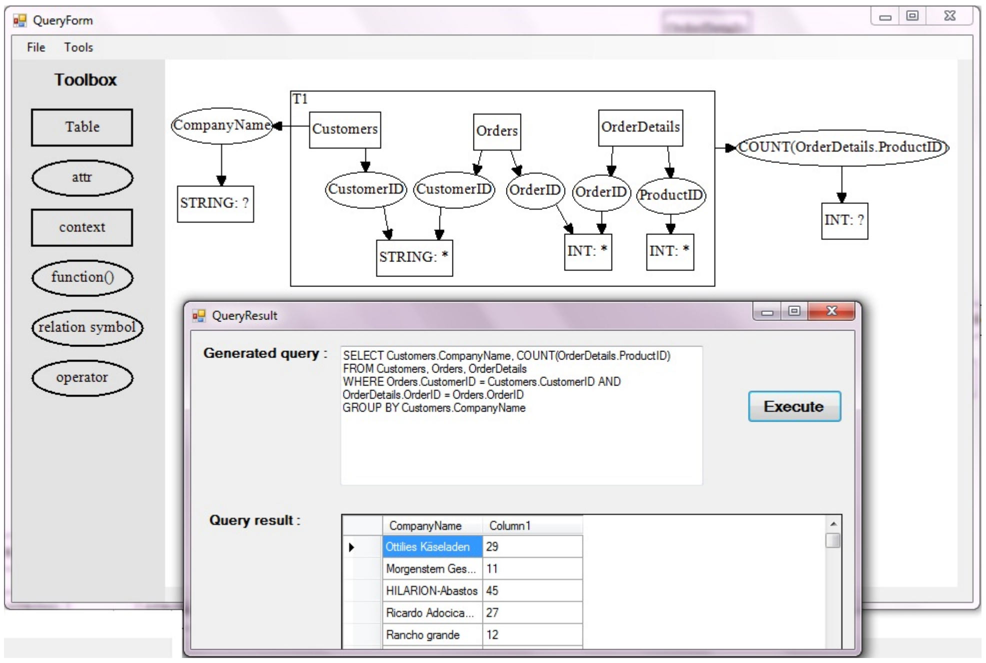
FIGURE 2. Conceptual graph for query involving join operation

concrete term, here $Y$, as the predicate. The object of discussion or thought thus introduced may also be called a hypostatic object ([9]).

In reasoning, we can consider the elements of a set and their properties, and sometimes the set itself as an element on its own having other properties. This shift in perspective transforms a relation into an object which then may be attached to relations again. In visualization, we use nested concept boxes as rectangle T1, which contains the joins of the tables: `Orders`, `OrderDetails` and `Products`. The result of the join is a relation too and the aggregation is applied to it.

## 3. Mining functional dependencies in relational databases

Functional dependencies (FDs shortly) are the most common integrity constraints encountered in databases. FDs are very important in relational database design to avoid data redundancy. Extracting FDs from a relational table is a crucial task to understand data semantics useful in many database applications. The subject of detecting functional dependencies in relational tables was studied for a long time and recently addressed with a data mining



```
SELECT Customers.CompanyName, COUNT(OrderDetails.ProductID)
FROM Customers, Orders, OrderDetails
WHERE Orders.CustomerID = Customers.CustomerID AND
OrderDetails.OrderID = Orders.OrderID
GROUP BY Customers.CompanyName
```

| CompanyName | Column1 |
|---|---|
| Ottilies Käseladen | 29 |
| Morgenstern Ges... | 11 |
| HILARION-Abastos | 45 |
| Ricardo Adocica... | 27 |
| Rancho grande | 12 |

FIGURE 3. Conceptual graph for query involving grouping and aggregate functions

viewpoint. Baixeries [2] gives an interesting framework to mine functional dependencies using Formal Context Analysis.

Hereth [7] presents how some basic concepts from database theory are translated into the language of Formal Concept Analysis. The definition of the formal context of functional dependencies for a relational table can also be found in [7]. Regarding to this definition, the context's attributes are the columns (named attributes) of the table, the tuple pairs of the table will be the objects of the context. In [7] you can find the proposition which asserts that in this context, implications are essentially functional dependencies between the columns of the relational database table.

A detailed analysis and complex examples of the formal context of functional dependencies for a relational table are presented in [15]. The novelty of our method is that it builds inverted index files in order to optimize the construction of the formal context of functional dependencies.

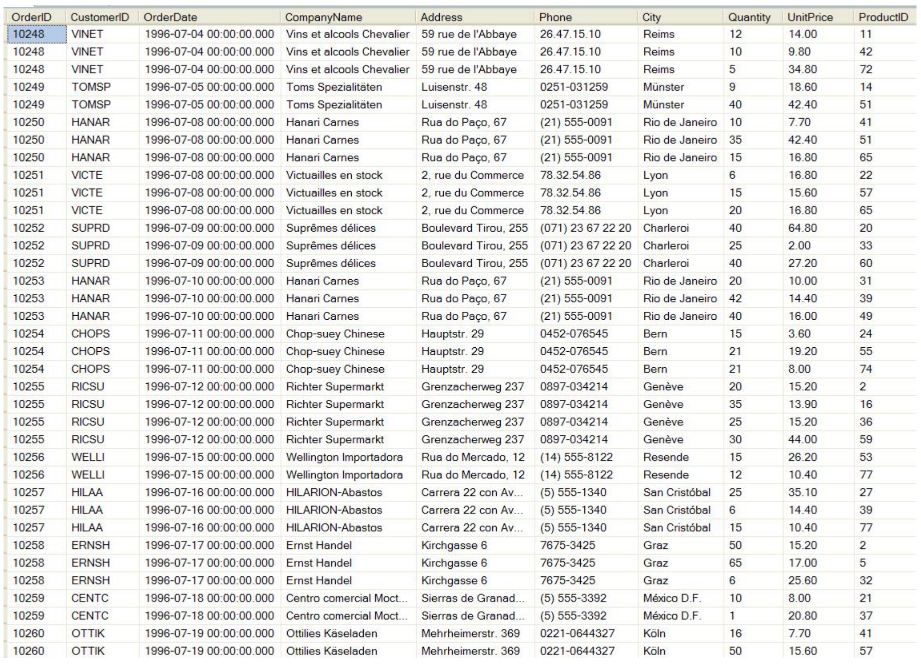
We implemented the method presented in [15] and completed it with a software tool, which analyzes an existing relational database table. Our software named FCAFuncDepMine (see [17]) can connect to a MS SQL Server, Oracle or MySQL database by giving the type and the name of the database, a login name and password, then the software offers a list of identified table names which can be selected for possible functional dependency examination.

It constructs the formal context of functional dependencies, uses Conexp [22] to build the concept lattice and to determine the implications in this context, which corresponds to functional dependencies in the analyzed table. The software can be used in relational database design and for detecting functional dependencies in existing tables, respectively. A detailed data analysis using software FCAFuncDepMine is presented in [16].

**Example 4.** Let be a complex example, which is illustrative for our work. The next table:

```
OrderDetail [OrderID, CustomerID, OrderDate, CompanyName,
    Address, Phone, City, Quantity, UnitPrice, ProductID]
```

stores orders of different customers together with order details information as product ID, order price and quantity too. `CompanyName` is the name of customer, `Address` is the customer's address and the `City` is his city. In Figure 4 there are the first rows from table `OrderDetail` .



| OrderID | CustomerID | OrderDate | CompanyName | Address | Phone | City | Quantity | UnitPrice | ProductID |
|---|---|---|---|---|---|---|---|---|---|
| 10248 | VINET | 1996-07-04 00:00:00.000 | Vins et alcools Chevalier | 59 rue de l'Abbaye | 26.47.15.10 | Reims | 12 | 14.00 | 11 |
| 10248 | VINET | 1996-07-04 00:00:00.000 | Vins et alcools Chevalier | 59 rue de l'Abbaye | 26.47.15.10 | Reims | 10 | 9.80 | 42 |
| 10248 | VINET | 1996-07-04 00:00:00.000 | Vins et alcools Chevalier | 59 rue de l'Abbaye | 26.47.15.10 | Reims | 5 | 34.80 | 72 |
| 10249 | TOMSP | 1996-07-05 00:00:00.000 | Toms Spezialitäten | Luisenstr. 48 | 0251-031259 | Münster | 9 | 18.60 | 14 |
| 10249 | TOMSP | 1996-07-05 00:00:00.000 | Toms Spezialitäten | Luisenstr. 48 | 0251-031259 | Münster | 40 | 42.40 | 51 |
| 10250 | HANAR | 1996-07-08 00:00:00.000 | Hanari Carnes | Rua do Paço, 67 | (21) 555-0091 | Rio de Janeiro | 10 | 7.70 | 41 |
| 10250 | HANAR | 1996-07-08 00:00:00.000 | Hanari Carnes | Rua do Paço, 67 | (21) 555-0091 | Rio de Janeiro | 35 | 42.40 | 51 |
| 10250 | HANAR | 1996-07-08 00:00:00.000 | Hanari Carnes | Rua do Paço, 67 | (21) 555-0091 | Rio de Janeiro | 15 | 16.80 | 65 |
| 10251 | VICTE | 1996-07-08 00:00:00.000 | Victuailles en stock | 2, rue du Commerce | 78.32.54.86 | Lyon | 6 | 16.80 | 22 |
| 10251 | VICTE | 1996-07-08 00:00:00.000 | Victuailles en stock | 2, rue du Commerce | 78.32.54.86 | Lyon | 15 | 15.60 | 57 |
| 10251 | VICTE | 1996-07-08 00:00:00.000 | Victuailles en stock | 2, rue du Commerce | 78.32.54.86 | Lyon | 20 | 16.80 | 65 |
| 10252 | SUPRD | 1996-07-09 00:00:00.000 | Suprêmes délices | Boulevard Tirou, 255 | (071) 23 67 22 20 | Charleroi | 40 | 64.80 | 20 |
| 10252 | SUPRD | 1996-07-09 00:00:00.000 | Suprêmes délices | Boulevard Tirou, 255 | (071) 23 67 22 20 | Charleroi | 25 | 2.00 | 33 |
| 10252 | SUPRD | 1996-07-09 00:00:00.000 | Suprêmes délices | Boulevard Tirou, 255 | (071) 23 67 22 20 | Charleroi | 40 | 27.20 | 60 |
| 10253 | HANAR | 1996-07-10 00:00:00.000 | Hanari Carnes | Rua do Paço, 67 | (21) 555-0091 | Rio de Janeiro | 20 | 10.00 | 31 |
| 10253 | HANAR | 1996-07-10 00:00:00.000 | Hanari Carnes | Rua do Paço, 67 | (21) 555-0091 | Rio de Janeiro | 42 | 14.40 | 39 |
| 10253 | HANAR | 1996-07-10 00:00:00.000 | Hanari Carnes | Rua do Paço, 67 | (21) 555-0091 | Rio de Janeiro | 40 | 16.00 | 49 |
| 10254 | CHOPS | 1996-07-11 00:00:00.000 | Chop-suey Chinese | Hauptstr. 29 | 0452-076545 | Bern | 15 | 3.60 | 24 |
| 10254 | CHOPS | 1996-07-11 00:00:00.000 | Chop-suey Chinese | Hauptstr. 29 | 0452-076545 | Bern | 21 | 19.20 | 55 |
| 10254 | CHOPS | 1996-07-11 00:00:00.000 | Chop-suey Chinese | Hauptstr. 29 | 0452-076545 | Bern | 21 | 8.00 | 74 |
| 10255 | RICSU | 1996-07-12 00:00:00.000 | Richter Supermarkt | Grenzacherweg 237 | 0897-034214 | Genève | 20 | 15.20 | 2 |
| 10255 | RICSU | 1996-07-12 00:00:00.000 | Richter Supermarkt | Grenzacherweg 237 | 0897-034214 | Genève | 35 | 13.90 | 16 |
| 10255 | RICSU | 1996-07-12 00:00:00.000 | Richter Supermarkt | Grenzacherweg 237 | 0897-034214 | Genève | 25 | 15.20 | 36 |
| 10255 | RICSU | 1996-07-12 00:00:00.000 | Richter Supermarkt | Grenzacherweg 237 | 0897-034214 | Genève | 30 | 44.00 | 59 |
| 10256 | WELLI | 1996-07-15 00:00:00.000 | Wellington Importadora | Rua do Mercado, 12 | (14) 555-8122 | Resende | 15 | 26.20 | 53 |
| 10256 | WELLI | 1996-07-15 00:00:00.000 | Wellington Importadora | Rua do Mercado, 12 | (14) 555-8122 | Resende | 12 | 10.40 | 77 |
| 10257 | HILAA | 1996-07-16 00:00:00.000 | HILARION-Abastos | Carrera 22 con Av... | (5) 555-1340 | San Cristóbal | 25 | 35.10 | 27 |
| 10257 | HILAA | 1996-07-16 00:00:00.000 | HILARION-Abastos | Carrera 22 con Av... | (5) 555-1340 | San Cristóbal | 6 | 14.40 | 39 |
| 10257 | HILAA | 1996-07-16 00:00:00.000 | HILARION-Abastos | Carrera 22 con Av... | (5) 555-1340 | San Cristóbal | 15 | 10.40 | 77 |
| 10258 | ERNSH | 1996-07-17 00:00:00.000 | Ernst Handel | Kirchgasse 6 | 7675-3425 | Graz | 50 | 15.20 | 2 |
| 10258 | ERNSH | 1996-07-17 00:00:00.000 | Ernst Handel | Kirchgasse 6 | 7675-3425 | Graz | 65 | 17.00 | 5 |
| 10258 | ERNSH | 1996-07-17 00:00:00.000 | Ernst Handel | Kirchgasse 6 | 7675-3425 | Graz | 6 | 25.60 | 32 |
| 10259 | CENTC | 1996-07-18 00:00:00.000 | Centro comercial Moct... | Sierras de Granad... | (5) 555-3392 | México D.F. | 10 | 8.00 | 21 |
| 10259 | CENTC | 1996-07-18 00:00:00.000 | Centro comercial Moct... | Sierras de Granad... | (5) 555-3392 | México D.F. | 1 | 20.80 | 37 |
| 10260 | OTTIK | 1996-07-19 00:00:00.000 | Ottilies Käseladen | Mehrheimerstr. 369 | 0221-0644327 | Köln | 16 | 7.70 | 41 |
| 10260 | OTTIK | 1996-07-19 00:00:00.000 | Ottilies Käseladen | Mehrheimerstr. 369 | 0221-0644327 | Köln | 50 | 15.60 | 57 |

FIGURE 4. First rows from table **OrderDetail**

The conceptual lattice for context of functional dependencies in table `OrderDetail` is represented in Figure 5.

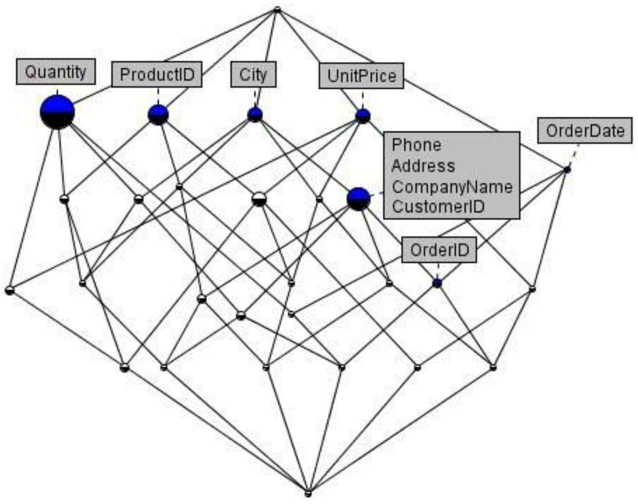
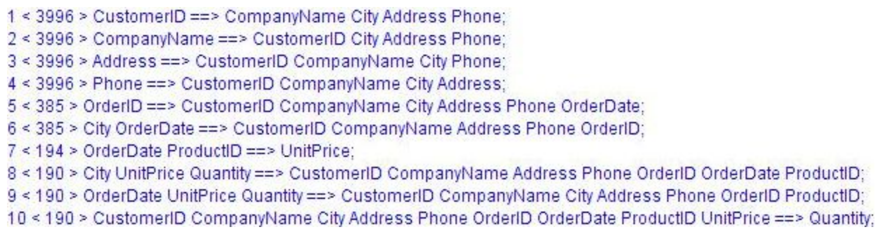


FIGURE 5. Conceptual lattice for context of functional dependencies in table **OrderDetail**

The implications in this lattice, which are functional dependencies in the table can be seen as follows: concept with label `OrderID` is subconcept of concept with labels `Phone, Address, CompanyName, CustomerID` and of concept with label `OrderDate` too. So we can read the following functional dependencies.

$$OrderID \rightarrow CustomerID, CompanyName, Address, Phone$$



```
1 < 3996 > CustomerID ==> CompanyName City Address Phone;
2 < 3996 > CompanyName ==> CustomerID City Address Phone;
3 < 3996 > Address ==> CustomerID CompanyName City Phone;
4 < 3996 > Phone ==> CustomerID CompanyName City Address;
5 < 385 > OrderID ==> CustomerID CompanyName City Address Phone OrderDate;
6 < 385 > City OrderDate ==> CustomerID CompanyName Address Phone OrderID;
7 < 194 > OrderDate ProductID ==> UnitPrice;
8 < 190 > City UnitPrice Quantity ==> CustomerID CompanyName Address Phone OrderID OrderDate ProductID;
9 < 190 > OrderDate UnitPrice Quantity ==> CustomerID CompanyName City Address Phone OrderID ProductID;
10 < 190 > CustomerID CompanyName City Address Phone OrderID OrderDate ProductID UnitPrice ==> Quantity;
```

FIGURE 6. Implications, namely functional dependencies in the table **OrderDetail**

Another implication is: $OrderID \rightarrow OrderDate$. The possible implications given by Conexp software are in Figure 6. The user can make attribute exploration to decide which implication is valid. The number before the implication can help us. The implications labeled with biger numbers usually are valid. Having these functional dependencies we can propose the decomposition of the table `OrderDetail`. It is clear that the information about customers have to be in a separate table, candidate keys is `Customers` table are: `CustomerID, CompanyName, Phone, Address`. If we introduce the same name for different customers, the `CompanyName` attribute will appear in a concept, which is subconcept of the concept with label `CustomerID, Phone, Address`. The same results if we introduce different companies with the same address. The `CustomerID` attribute is functionally dependent on `OrderID`, so we will design an `Orders` table too. The proposed tables are:

```
Customers(CustomerID, CompanyName, City, Address, Phone)
Orders(OrderID, CustomerID, OrderDate)
OrderDetails(OrderID, ProductID, UnitPrice, Quantity)
```

## 4. Future work

There are two directions we intend to apply conceptual knowlegde processing. We are developing the mathematical backgound for the schema and query model with conceptual graphs of a relational database.

On the other hand, we intend to analyze semistructured data presented in XML form. Many papers are dedicated to the theory of functional dependency in XML data. Our aim is to construct the context of functional dependencies for an XML tree and generate implications in it, which will be functional dependencies in XML data.

## References

[1] Abiteboul, S., Hull, R., Vianu, V.: *Foundations of Databases.* Addison-Wesley, Reading - Menlo - New York (1995)

[2] Baixeries, J.: *A Formal Concept Analysis Framework to Mine Functional Dependencies*, Proceedings of Mathematical Methods for Learning, (2004).

[3] Carpineto, C., Romano, G.: *Concept Data Analysis, Theory and Applications*, Wiley and Sons, 2004.

[4] Dau, F.: *The Logic System of Concept Graphs with Negation And Its Relationship to Predicate Logic*, LNCS, Vol. 2892, Springer Berlin / Heidelberg (2003)

[5] Dau, F., Hereth, J. C.: *Nested Concept Graphs: Mathematical Foundations and Applications for Databases.* In: Ganter, B.; de Moor, A. (eds.): Using Conceptual Structures. Contributions to ICCS 2003. Shaker Verlag, Aachen, (2003), pp. 125-139.

[6] Devlin, K.: *Infosense - Turning Information into Knowledge*, Freeman, New York, 1999.

[7] Hereth, J.: *Relational Scaling and Databases.* Proceedings of the 10th International Conference on Conceptual Structures: Integration and Interfaces LNCS 2393, Springer Verlag (2002) pp. 62-76

[8] Boksenbaum, C., Carbonneill, B., Haemmerle O., Libourel, T.: *Conceptual Graphs for Relational Databases* in Conceptual Graphs for Knowledge Representation., Guy, M. W., Moulin B., Sowa, J. F. eds. Lecture Notes in AI 699, Springer-Verlag, Berlin (1993).

[9] Peirce, C.S.: *The Simplest Mathematics*, in Collected Papers, CP4.235, CP4.227-323, 1902.

[10] Silberschatz, A., Korth, H. F.,Sudarshan, S.: *Database System Concepts*, McGraw-Hill, Fifth Edition, (2005)

[11] Sowa, J. F.: *Conceptual Graphs for a Database Interface.* In: IBM Journal of Research and Development, vol. 20, no. 4, (1976) pp. 336-357.

[12] Sowa, J. F.: *Conceptual Structures: Information Processing in Mind and Machine.* Addison Wesley Publishing Company Reading, (1984).

[13] Sowa, J. F.: *Conceptual graphs summary*, in Nagle, T. E.; Nagle, J. A.; Gerholz, L. and Eklund, P. W. (editors): Conceptual Structures: Current Research and Practice, Ellis Horwood, (1992), pp 3-51.

[14] Sowa, J. F.: *Knowledge Representation: Logical, Philosophical, and Computational Foundations*, Brooks Cole Publishing Co., Pacific Grove, CA. (2000)

[15] Janosi Rancz, K. T., Varga, V.: *A Method for Mining Functional Dependecies in Relational Database Design Using FCA*, Studia Univ. Babeş-Bolyai, Informatica, Vol. LIII, Nr. 1(2008).

[16] Janosi Rancz, K.T., Varga, V., Puskas, J.: *A Software Tool for Data Analysis Based on Formal Concept Analysis*, Studia Univ. Babeş-Bolyai, Informatica, Vol. LIII, Nr. 2(2008).

[17] Varga, V., Janosi Rancz, K. T.: *A Software Tool to Transform Relational Databases in order to Mine Functional Dependencies in it Using Formal Concept Analysis*, Proc. of the Sixth International Conference on Concept Lattices and Their Applications, Olomouc, 21-23 October, 2008. pp. 1-8.

[18] Varga, V., Săcărea, C., Takács, A.: *A Software Tool for Interactive Database Access using Conceptual Graphs*, International Conference Knowledge Engineering Principles and Techniques, KEPT 2009, Cluj-Napoca, July 2-4.

[19] Wille, R.: *Conceptual Graphs and Formal Concept Analysis*, Lecture Notes In Computer Science; Vol. 1257, Proceedings of the Fifth International Conference on Conceptual Structures: Fulfilling Peirce's Dream, Springer Verlag (1997), pp 290 - 303.

[20] Wille, R.: *Conceptual Contents as Information - Basics for Contextual Judgement Logic*, Conceptual Structures for Knowledge Creation and Communication, ICCS 2003, LNAI 2746, Springer, pp. 1-15, 2003.

[21] Wille, R.: *Methods of Conceptual Knowledge Processing*, ICFCA 2006, LNAI 3874, Springer, pp. 1-29, 2006.

[22] Serhiy A. Yevtushenko: *System of Data Analysis "Concept Explorer"*. (In Russian). Proceedings of the 7th National Conference on Artificial Intelligence KII-2000, p. 127-134, Russia, 2000.

Babes-Bolyai University, Faculty of Mathematics and Computer Science, Cluj-Napoca, Romania

*E-mail address*: `csacarea@math.ubbcluj.ro`

*E-mail address*: `ivarga@nessie.cs.ubbcluj.ro`