Faculty of Computer Science, Free University of Bozen-Bolzano, Piazza Domenicani 3, 39100 Bolzano, Italy
Tel: +39 04710 16000, fax: +39 04710 16009, http://www.inf.unibz.it/krdb/

KRDB Research Centre Technical Report:

Querying Structured Data with Lite Natural Language

Raffaella Bernardi, Diego Calvanese, Camilo Thorne

Affiliation	KRDB Research Centre, Faculty of Computer Science Free University of Bozen-Bolzano Piazza Domenicani 3, 39100, Bolzano, Italy
Corresponding author	Camilo Thorne cthorne@inf.unibz.it
Keywords	Natural language interfaces to databases, controlled languages, formal semantics, conjunctive queries
Number	KRDB08-06
Date	1-7-08
URL	http://www.inf.unibz.it/krdb/pub/

Querying Structured Data with Lite Natural Language

Raffaella Bernardi, Diego Calvanese, Camilo Thorne

KRDB Research Centre
Free University of Bozen-Bolzano
3, Piazza Domenicani, 39100 (Italy)
{bernardi,calvanese,cthorne}@inf.unibz.it

Abstract. Querying and managing structured data in natural language is a challenging task due to its ambiguity (syntactic and semantic) and its expressiveness. On the other hand, querying, for example, a database is a well-defined and unambigous task, namely, that of evaluating some formal query of a limited expressiveness over the instance of a database schema. This problem can be tackled, we believe, by defining a controlled fragment of English, Lite English, whose declarative sentences compositionally translate into a description logic, DL-Lite, and whose questions likewise translate into a simple relational query language, conjunctive queries (CQs). The query answering problem (QE) for DL-Lite knowlege bases is equivalent to that of relational databases, a property preserved by Lite English. Lite-English is thus capable of performing the same tasks DL-Lite can accomplish (declaring, specifying and querying data), but in a manner more intuitive to the end-user.
Keywords Controlled Language, Conjuctive Queries, Formal Semantics, Description Logics.

1 Motivation

Relational database management systems (RDBMS) like dBASE or Oracle, are an attempt to the tasks of structuring, modelling, declaring, updating and querying data. The interfaces of these systems are based on formal query languages that combine both declarative and imperative features such as SQL (cf. [6]). Crucially, the expressive power of these formal languages should be well-known and well defined, since the problem of query answering (QE) for relational databases (DBs) should be **LOGSPACE** on *data complexity*, i.e., on the number of records of the database.

But using these languages requires some previous training and can prove counterintuitive to the a casual end user. For such a user the intuitive appeal and understanding of the machine interface can be crucial. It would thus be suitable in such cases to shift to natural language (NL) and to use natural language questions instead of formal queries

We believe that to address this problem a compromise between expressive power and the intuitive appeal of NL has to be reached and will argue further that this compromise involves the use of the so-called *controlled languages*

(CLs), which are fragments of NL tailored to deal with these tasks and where utterances compositionally translate into a logical expression called *meaning representation* (MR), that encodes semantics at the sentence level (cf. [8]). Since we are interested in QE, we will focus in NL questions.

The structure of this paper is as follows. Section 2 will briefly recall QE for relational databases, that is, for the relational data model (RDM). Section 3 will introduce a description logic, DL-Lite, which will be our MR formalism (and ecode the semantics of our utterances) and will argue why this is a good choice. Section 4 will recall some basic properties of NL questions. In Section 5 we will proceed to define the controlled language, Lite English, that is associated to this logic and say how Lite English deals with questions. The suitability of Lite English will be validated in Section 6, by looking at a small corpus of questions. Finally, Section 7 will sum up our results and conclusions and make clear how we intend to extend our work in the future.

2 Relational Databases and QE

In this section we recall the basics of the RDM. A *database schema* $\mathbf{R} = \{R_1, ..., R_n\}$ is a finite set of relation names with an associated *arity* (an integer $n \geq 0$). Next, let **Dom** be a (possibly) countably infinite non-empty set of constants called *domain*, then: a *tuple* over **Dom** is every $\boldsymbol{t} \in \bigcup_{i \in \mathbb{N}} \mathbf{Dom}^i$ and a *relation instance* of a relation name R of arity n is a relation $R(\mathbf{I}) \subseteq \mathbf{Dom}^n$ – hence, a set of tuples. Finally, a *database instance* $\mathbf{I}$ over schema $\mathbf{R}$ is a finite set of relation instances and the *active domain* of instance $\mathbf{I}$ is the finite set $adom(\mathbf{I}) \subsetneq \mathbf{Dom}$ of (pairwise distinct) domain constants occuring among the relations in $\mathbf{I}$.

The intuition behind the RDM is that a database schema $\mathbf{R}$ can be seen as a set of relation constants from a first order signature and its instance $\mathbf{I}$ as one of its corresponding first order finite interpretation structures (cf. [6]). Accordingly, formal query languages will be defined following first order logic (FOL) or some fragment of it. Of these, perhaps the most well known is that of conjunctive queries:

Definition 1. (Conjunctive Queries) *A* conjunctive query *(CQ) over schema* $\mathbf{R}$ *is an expression of the form:*

$$q(\boldsymbol{x}) \leftarrow R_1(\boldsymbol{y}_1), ..., R_n(\boldsymbol{y}_n).$$

Where $R_i \in \mathbf{R}$*, for* $i \in [1, n]$*,* $\boldsymbol{x}$ *is a possibly empty finite sequence of* distinguished variables *and the* $\boldsymbol{y}_i$*'s, for* $i \in [1, n]$*, are finite sequences of variables and constants. Furthermore, the variables of* $\boldsymbol{x}$ *must occur among the* y_i*, for* $i \in [1, n]$ *(*safeness*).*

The symbols on the left hand side of $\leftarrow$ are called the *head* of the query and those on its right hand side its *body*. The body can be seen as a conjunction of atoms, with the comma representing logical conjunction. The variables in the body that are not distinguished are assumed to be existentially quantified. If the sequence of distinguished variables is empty, the query is called a *boolean*

TBox

$MarriedMan \sqsubseteq Man$
$Man \sqsubseteq \exists Loves$
$Man \sqsubseteq \exists Loves : Woman$
$Man \sqsubseteq LivingCreature$

ABox

$MarriedMan(Julian)$
$Loves(Julian, Mary)$

Fig. 1. A sample DL-Lite KB, KB_0.

query. Note that there is no negation symbol, no disjunction and no universal quantification and that hence CQs constitute a proper fragment of FOL (in particular, we lack a complete set of boolean operators). The *semantics* of a CQ q, denoted $q(\mathbf{I})$, is given by the set of tuples that satisfy her in the DB instance $\mathbf{I}$. QE is the problem of computing this particular set and this can be done on space logarithmic on the size of the database, that is on $\#(adom(\mathbf{I}))$, a.k.a. its *data complexity*:

Proposition 1. ([6]) *QE for relational databases is* **LOGSPACE** *on data complexity.*

To sum up, there are two key conditions to be satisfied whenever we want to access data stored in a relational database:

(i) A CQ q must characterize exactly the data to be retrieved – the set of tuples that satisfy query q w.r.t. database instance $\mathbf{I}$.
(ii) QE has to be **LOGSPACE** on data complexity, that is, on the number of tuples of the database.

3 DL-Lite and QE

Description logics (DLs) bring a certain number of advantages to data management and access. DLs are decidable fragments of FOL that constitute, to a great extent, the logical underpinning of the ontology languages, like OWL, which serve to provide conceptual models of data domains (cf. [2]). DL theories are called *knowlege bases* and they typically comprise a set of universally quantified assertions, the ontology that models the data domain and a set of grounded atoms that model the data.

But which is the optimal (or maximal) DL for carrying on with these tasks? The answer is: DL-Lite (cf. [2]). This is critical for a CL approach, because a CL needs, for its semantics, to compositionally translate into a logic. Why? Because we can encode databases and their conceptual models with logical theories (sets of assertions) known as *knowledge bases* and then formally characterize QE in terms of logical entailment, as the reader shall see below. This latter feature, in particular, adds a reasoning layer to RDMSs by means of a so-called ontology-driven data access (cf. [2]) and thus provides a framework for extending the functionalities of DBMSs. But then, since its main task and goal is that of managing data by way of relational DBs, it has to satisfy the two conditions that the RDM imposes on us. As we shall see, DL-Lite has been carefully tailored to fit them.

Furthermore, DL-Lite can be seen as the maximal tractable description logic (DL) capable of expressing most of database constraints (the fundamental features of ER-diagrams can be mapped onto DL-Lite concepts and assertions). Moreover, it can be conceived as a *decidable* and *tractable* fragment of FOL (satifiability for TBoxes is in **P**).

Definition 2. (Concepts) *Let* $\mathcal{P} = \{P_i | i \in \mathbb{N}\}$ *and* $\mathcal{R} = \{R_i | i \in \mathbb{N}\}$ *be two countable sets of primitive concept and role symbols. DL-Lite* left hand side concepts $\mathcal{B}$ *and* right hand side concepts $\mathcal{C}$ *are defined as follows:*

1. $\mathcal{B} ::= \mathcal{P} \mid \exists\mathcal{R} \mid \exists\mathcal{R}^- \mid \mathcal{B} \sqcap \mathcal{B}$.
2. $\mathcal{C} ::= \neg\mathcal{P} \mid \neg\exists\mathcal{R} \mid \neg\exists\mathcal{R}^- \mid \mathcal{B} \mid \mathcal{C} \sqcap \mathcal{C} \mid \exists\mathcal{R} : \mathcal{C} \mid \exists\mathcal{R}^- : \mathcal{C}$.

Thinking in terms of the well-known ER diagram formalism, concepts can be styled formal counterparts of entities (or classes), representing collections of individuals, and roles, as binary associations linking entities and thus holding over the individuals belonging to the class (cf. [2]). Next, assertions:

Definition 3. (Assertions) *Let* $\mathcal{K} = \{c_i | i \in \mathbb{N}\}$ *be a set of constants. DL-Lite* facts $\mathcal{A}$ *and* terminological assertions $\mathcal{T}$ *are defined as follows:*

1. $\mathcal{A} ::= \mathcal{B}(\mathcal{K}) \mid \mathcal{R}(\mathcal{K}, \mathcal{K}) \mid \mathcal{C}(\mathcal{K})$.
2. $\mathcal{T} ::= \mathcal{B} \sqsubseteq \mathcal{C}$.

So, right hand side concepts occur at the right hand side of the inheritance or inclusion relation symbol $\sqsubseteq$. Terminological assertions are assumed to be implicitely universally quantified. Facts correspond to ground atoms.

Definition 4. (Knowledge Bases) *A DL-Lite* knowledge base *(KB) is a tuple* $KB = \langle ABox, TBox \rangle$*, where the ABox is a set of facts and the TBox a set of terminological assertions.*

The ABox is also known as the *extensional knowledge base* and the TBox as the *intensional knowledge base*. The intensional knowledge base encodes, intuitevely, the ontologies or conceptual models (the specification) and the extensional knowledge base the actual data to be declared or stored. The *size* of KB is given by the number of pairwise distinct constants of its ABox. See Figure 1 for an example of a KB of size 3. To retrieve data from a KB we consider CQs with the following caveat: they are to be built over the basic concept language of DL-Lite. That is, both queries and DL-Lite assertions (terminological or facts) are built over the same signature. CQs can be used to retrieve data from DL-Lite KBs. QE is modelled in terms of logical *entailemnt*: the semantics of a CQ q in this context is given by the set of constant sequences $\boldsymbol{c}$ such that the logical entailment $\langle \text{ABox}, \text{TBox} \rangle \models q'$, where q' denotes the grounding of q w.r.t. $\boldsymbol{c}$, holds. Computing this set is logarithmic on the KB's size (i.e., its *data complexity*):

Proposition 2. (Calvanese *et. al.* [2]) *QE for DL-Lite KBs is* **LOGSPACE** *on data complexity.*

4 NL Questions

Natural language questions can be divided into many kinds. We will concentrate on solely Wh-questions since their semantics bear many similarities with that of CQs (cf. [3, 4]). English *Wh-questions* are formed by combining a wh-word, a pronoun in syntactic terms, like "how", "what", "which" or "who". The former two behave in a way similar to a generalized determiner, the latter two as a generalized quantifier (cf. [3]). Thus we can divide them into two different subclasses, following the schema below:

$$\text{Type (i)} = \begin{cases} \text{what N VP} \\ \text{how Adj VP} \end{cases}$$
$$\text{Type (ii)} = \begin{cases} \text{who/which Aux VP} \\ \text{who/which VP} \end{cases}$$

Now, their semantics. Following the higher order logic (HOL) formal compositional semantics paradigm (cf. [3, 4]), a *Wh-question operator* is an expression $Q_{wh} \colon (\tau \to t) \to (\tau \to t)$. As usual when speaking about NL semantics, we restrict ourselves to the case where $\tau := e$, following Carpenter and Clifford (cf. [3, 4]). This operator is applied to a verb phrase of type $e \to t$, called the *body* of the question, giving way to a question. Questions are expressions of type $e \to t$, i.e., they denote a set of individuals, namely those they characterize – in a way much similar to that of CQs. Type e is formally associated, as usual, to the domain D_e of *individuals* of a HOL denotational frame and t to the set $D_t = \{0, 1\}$ of boolean *truth values*. Applying a "how" to an adjective or a "what" to a noun yields a Wh-question operator. The table below recalls the lexical (HOL) semantics of interrogative pronouns:

Wh-word	**MR**
who/which	$\lambda P.\lambda x.(P)x \colon (e \to t) \to (e \to t)$
what/how	$\lambda P.\lambda Q.\lambda x[(P)x \to (Q)x] \colon (e \to t) \to ((e \to t) \to (e \to t))$

5 Lite English

But then, how to link Wh-questions to CQs? And moreover, to CQs over DL-Lite KBs? Recall that the body of a CQ in this latter case is built as a conjunction out of atoms involving unary and binary predicates and free variables subject exclusively to existential quantification. Recall that what we want is to be able to declare and specify information strcutured as a KB or as a DB on top of which we may have added an ontology (a TBox) in NL and then query it using CQs. Moreover, the complexity bounds on this task have to be respected. As we already said, we will restrict our analysis to Wh-questions and for the sake of simplicity will focus on questions whose NPs are all singular and whose VPs and *a fortiori* their verbs are inflected in the 3rd person singular of the active voice. This means that the words we can afford in our lexicon, if we want to compositionally translate our questions into CQs, are basically these:

Function Lexicon	**Content Lexicon**
some	PNs
somebody	Adjs
is a/ is	Ns
and	TVs
who/that	IVs
which/what/who/how	
does, could,...	

Because only questions whose syntactic constructs express conjunction (relatives, conjunction) and existential quantification can or may express a CQ. Negation and universal quantification are to be banned. For each content word category we may allow an arbitrarily large number of tokens. Note that our function lexicon may contain an otherwise finite number auxiliary and modal verbs like ”does”, ”could”, ”can”, etc. The words of these table Because these are basically the only words whose lexical semantics as stated in HOL formal semantics (cf. [3]), encode in NL the logical constructs and operations CQs allow.

To this effect we are developing a controlled language, Lite English, divided in two parts: **(i)** The declarative fragment that compositionally translates into DL-Lite assertions and: **(ii)** The interrogative fragment that translates into CQs. In this paper we will not speak about the the declarative fragment, for which we send the reader to [1] and [9].

To begin, we would like to remark that there are some syntactic constructs that we can safely paraphrase into one of the two main wh-question types we spoke before, which basically allow to ”define out” by some kind of contextual definition the definite article in contexts like the following:

(1) List the N/all the Ns that VP? $\rightsquigarrow$ Which N VP?

(2) Name the N/all the Ns that VP? $\rightsquigarrow$ Which N VP?

(3) What is the N/are the Ns that VP? $\rightsquigarrow$ What N VP?

(4) Give me the N/all the Ns that VP? $\rightsquigarrow$ Which N VP?

(5) Which is the N/are the Ns that VP? $\rightsquigarrow$ Which N VP?

(6) Show me the N/all the Ns that VP? $\rightsquigarrow$ Which N VP?

(7) Can/could you tell me the N/Ns that VP? $\rightsquigarrow$ Which N VP?

(8) Can/could you tell me what is the N that VP $\rightsquigarrow$ Which N VP?

Among other similar intuitively meaning-preserving paraphrases, Lite English thus only admits type (i) and (ii) wh-questions as previously schematized. It is only these that are subject to a formal semantic analysis.

This is quite a nice property, since the MR for "the" is the HOL expression (a generalized determiner) of the form $\lambda P.\lambda Q.\exists x[[(P)x \wedge (P)y] \wedge (\lambda Q.\exists x \forall y[(P)x \leftrightarrow x = y])P] : (e \to t) \to ((e \to t) \to t)$ (cf. [3]) which contains a universal quantifier and a relation, identity, we do not want in a CQ over a DL-Lite KB. Lite English thus provides rules only for the questions to the right of the $\leadsto$.

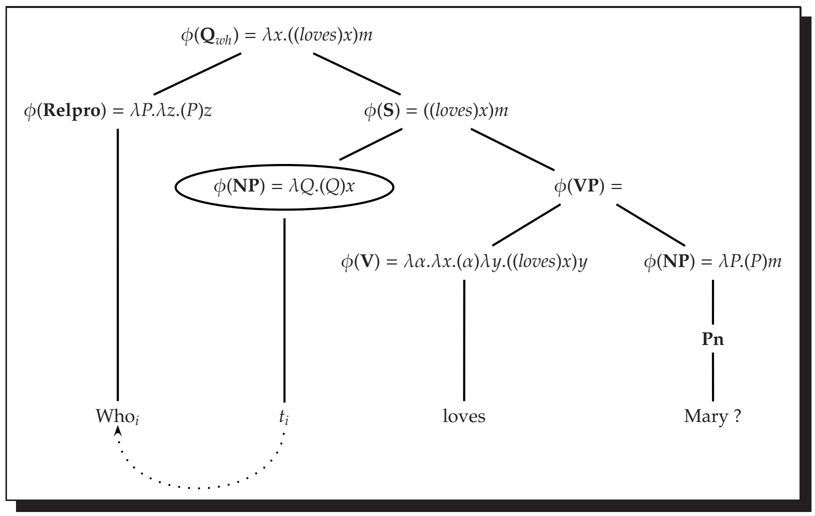

Fig. 2. Parsing "Who loves Mary?".

By a compositional translation we mean that HOL or type theoretical expressions are associated to the words both of the function and of the content lexicon and that the composition of syntactic components in parse tree is mirrored by lambda application and reduction (cf. [3, 4]), yielding ultimately a CQ meaning representation at the sentence level, when dealing with questions.

An example of a parse can be seen in Figure 2 above. What we compute in the end (the root node) is a well-typed HOL expression (namely, $\lambda x.((loves)x)m : e \to t$) which can be directly mapped to the CQ $q(x) \leftarrow loves(x, Mary)$, which, when evaluated over the KB KB_0 of Figure 1 above, returns the set $\{\langle Julian \rangle\}$, since $KB_0 \models q(Julian) \leftarrow loves(Julian, Mary)$, where $q(Julian) \leftarrow loves(Julian, Mary)$ is the grounding of $q(x) \leftarrow loves(x, Mary)$ obtained by the closed substitution $[Julian/x]$. At every node of the tree, we see the result of applying beta reduction, i.e., the current value of the compositional translation ϕ (cf. [3, 4]). We omit the (semantic) types for reasons of space. The tree contains a trace of whmovement yielding a gap-filler dependency. The circle around the NP indicates the boundaries of the gap (or island).

6 The Geobase Corpus

In order to validate the results regarding the expressive power of Lite English's questions, we manually examined a small corpus of questions, the Geobase corpus, which is a collection of 880 NL questions to a geography DB of the USA[1]. Figure 4 below gives a snapshot of the corpus. Figure 3 shows the DB schema. A double undelining indicates a primary key, while a simple one, a foreign key. We note that this DB is set within the *named perspective* of the RDM, which is otherwise equivalent to the unnamed perpective adopted in this paper (cf. [6]).

Our main goal in doing so was to see to what extent a *question* to a DB can be expressed or not by a CQ. This is important, since going beyond CQs takes QE over DL-Lite KBs well beyond the complexity of QE for BDs and hence prevent us from efficiently adding a reasoning layer on top of RDMSs, let alone a Lite English or CL layer. The main advantage is that this corpus focuses on questions to a DB and thus gives insight on the way a user may intend to retrieve data from a structured knowledge source.

To start, what about the the mismatch between DB relations and NL? If verbs are what convey relations in NL, then we cannot go beyond, maybe, 3-ary verbs (i.e., distransitive verbs), which means that we have to find some sensible way of paraphrasing this *n*-ary relations in NLs, let alone in CLs built out of them. But this is not so much of a problem since we can split DB relations into relations of lesser arity by means of *reification* (cf. [6]), a well-know technique within the DB community. But of course, we can only consider unary and binary relations in doing so, because DL-Lite and hence CQ relations are unary or binary. This is why we ruled out distransitive verbs from the content lexicon of Lite English, as already remarked.

Hence, the questions we should ask yoursleves are: When and how frequently can a question express a CQ and is *a fortiori* captured by Lite English?

[1] Available on the internet at ftp://ftp.cs.utexas.edu/pub/mooney/nl-ilp-data/geosystem.

State[StateName, Abbrev, Capital, Population, Area, Number, ListOfCities]
City[StateName, Abbrev, Name, Population]
River[RiverName, Length, ListOfStates]
Border[StateName, Abbreviation, ListOfStates]
Highlow[StateName, Abbrev, HighPoint, HighElev, LowPoint, LowElev]
Mountain[StateName, Abbrev, Name, Height]
Road[Number, ListOfStates]
Lake[Name, Area, ListOfStates]

Fig. 3. Geobase DB schema.

(i)	What is the capital of Georgia?
(ii)	What is the largest river in Washington state?
(iii)	How long is the Missouri River?
(iv)	Which states border Arizona?
(v)	What is the population of Dallas?
(vi)	Which state has the lowest population density?
(vii)	Which are the cities in Alaska?

Fig. 4. Sample Geobase corpus questions.

When and how often do NL questions go beyond this class of queries? Well, to start with, we can say when question from the Geobase corpus *cannot* express a CQ. In the corpus we see no occurrences of universal quantifiers or of words (determiners like ″every″ or ″no″, pronouns like ″everything″) associated to them. Nor are there any DTVs to be seen. In other words:

- Questions containing a negation (i.e., ″no″ or ″not″) cannot express a CQ.
- Questions that contain a definite article (″the″) that cannot be eliminated by a contextual definition as we did before (rules (1) to (8)) cannot express a CQ, because their semantics contains implicitly a universal quantifier (cf. [3] and the discussion above).
- Questions containing superlatives and comparatives, or counting expressions cannot express a CQ, since these constructs call for aggregation functions

The Geobase corpus shows that most of the user questions (and *a fortiori* queries) involve *aggregation* operations, conveyed by expressions like ″how many″, ″the average″, etc, i.e., superlative, counting words, comparatives and so forth, with, eventually, definite articles. For instance, question **(vi)** in Figure 4, that asks for the city with the minimal population density (i.e., the ratio of inhabitants per km^2) expresses a CQ with *aggregation functions* (CQs+agg) – a non conservative extension of CQs (and of TCs) where no function symbols were admitted in the language:

$$q(n, min(\frac{p}{a})) \leftarrow StateName(n, x, y, p, a, z, w). \tag{9}$$

This means that to actually determine whether a question in this corpus expresses or not a CQ we need only to focus on whether there are negations or aggregations in it.

As the following tables show, questions expressing CQs constitute roughly 1/3 of the total number of questions:

	CQs	**Aggregations**	**Negation**	**Total**
Wh-questions	304	575	1	880

	CQs	Aggregations	Negation
Wh-questions	34.54%	65.35%	0.11%

	CQs	Aggregations	Negation	Total
Type (i)	176	329	1	506
Type (ii)	128	246	0	374

	CQs	Aggregations	Negation
Type (i)	34.78%	65%	0.22%
Type (ii)	34.22%	65.78%	0%

The reason for this ubiquity of aggregations is the prsence of numerical data in the DB – which is quite common if not practically unavoidable in DBs by the way. It is thus a quite common phenomenon. However there are ways of ”defining out” aggregations: CQs and CQs+agg may collapse into one class. In effect, we can express an aggregation function like ”the highest” by computing, once again, a view (i.e., an auxiliary table), by first sorting the original table and then stroring the first record in the view. This is quite a common DB practice: a DB user wouldn't like to compute at runtime such a record when the DB contains tens of thousands of records as it may be the case for some of them. This means that all but one of the Geobase questions express a CQ and that they thus fall under Lite English or that they can be covered at any rate by this CL which, as the reader may recall, has been engineered so that its Wh-questions compositionally translate into CQs.

7 Conclusions and Future Work

In this paper we have characterized the main conditions any controlled language approach to structured data management (by way of a relational DB) must satisfy. In doing so, we have stressed the main tasks linked to data management and access in the RDM, together with the expressivity bounds linked to them, showing the impact that this has on the formal query languages supported by RDMSs. We have shown also how the DL-Lite description logic enhances these tasks, while preserving these expressivity bounds, and defined a controlled fragment of English, Lite English, that mirrors DL-Lite in NL, thus sharing its properties w.r.t. QE.

Lite English's Wh-questions compositionally translate into exactly CQs. Moreover, the suitability and intuitive appeal of such a fragment of English has been validated by looking at a corpus of questions to a geographical database, Geobase, that, as we have seen, express CQs. However, this result should perhaps be taken *cum grano salis*, given the low frequency of negations in these questions (0.22%) and the absence of universally quantified expressions and disjunction, all of which are quite natural in English. We would therefore like analyse by manual or automatical means larger corpora of questions, if possible for the final version of this paper to better validate Lite English and CL approaches.

Another thing to be done, again possibly for the final version of this paper is to stress the fact that CQs call for long distance gap-filler dependencies among main and subordinate clauses.

A point that we plan to study in the future is that of the monotonicity of questions. Indeed, we believe (but haven't proved as yet) that those NL questions whose type-theoretical semantics is monotone increasing (cf. [5]), can be mapped to conjunctive queries compositionally. Eventually, we would like to look at the kind of semantic properties different fragments of questions may characterize and compare them to the semantics of other formal query languages, in a manner similar to that of Third and Pratt w.r.t. the fragments of English (cf. [7]). These results in their turn should be validated by looking at larger question datasets.

References

1. Raffaella Bernardi, Diego Calvanese, and Camilo Thorne. Lite Natural Language. In *Proceedings of the 7th International Workshop on Computational Semantics (IWCS-7)*, 2007.
2. Diego Calvanese, Giusseppe de Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Data Complexity of Query Answering in Description Logics. In *Proceedings of the 10th International Conference on the Principles of Knowledge Representation and Reasoning (KR 2006)*, 2006.
3. Bob Carpenter. *Type-Logical Semantics*. The MIT Press, 1997.
4. James Clifford. Natural Language Querying of Historical Databases. *Computational Linguistics*, 14:10–35, 1988.
5. L. T. F. Gamut. *Logic, Language and Meaning (2 vols.)*. University of Chicago Press, 1991.
6. R. Hull, S. Abiteboul, and V. Vianu. *Foundations of Databases*. Addison-Welsey, 1995.
7. Ian Pratt and Allan Third. More Fragments of Language. *Notre Dame Journal of Formal Logic*, 2005.
8. John Sowa. *Knowledge Representation: Logical, Philosophical and Computational Foundations*. Brooks Cole Publishing Co., 1999.
9. Camilo Thorne. Controlled English for DL-Lite. Technical report, KRDB Research Centre, Free University of Bozen-Bolzano, 2007. http://www.inf.unibz.it/krdb/pub/index.php.