Automated Question Answering Using Question Templates that Cover the Conceptual Model of the Database

Eriks Sneiders

Department of Computer and Systems Sciences, Stockholm University and the Royal Institute of Technology, Forum 100, SE-164 40 Kista, Sweden
eriks@dsv.su.se

Abstract. The question-answering system developed by this research matches one-sentence-long user questions to a number of question templates that cover the conceptual model of the database and describe the concepts, their attributes, and the relationships in form of natural language questions. A question template resembles a frequently asked question (FAQ). Unlike a static FAQ, however, a question template may contain entity slots that are replaced by data instances from the underlying database. During the question-answering process, the system retrieves relevant data instances and question templates, and offers one or several interpretations of the original question. The user selects an interpretation to be answered.

1 Introduction

[1] groups natural language interfaces into pattern-matching systems, syntax-based systems, semantic grammar systems, and systems with intermediate query representations. The latter combines features of both syntax-based and semantic grammar systems. This short paper describes a prototype of a pattern-matching system, further called *question assistant*, which was first applied to the database on events in Stockholm, the capital of Sweden. The main advantage of a pattern matching system is its simplicity: no sophisticated processing of user questions is needed. The simplicity becomes essential for automated question answering on WWW because many small and medium-size websites cannot afford complex solutions that require much time and rare human skills to install and maintain the system.

A much broader description of this research can be found in [2].

2 Question Templates

The initial framework of this research was automated FAQ answering [3]. FAQ collections are created only for knowledge domains where concepts have rather few instances. No one is likely to write a separate FAQ for each data instance in a large

database. We can, however, benefit from the simplicity of automated FAQ answering in the task of creating a question-answering interface for a structured, e.g. relational, database.

Let us introduce a *question template* – a dynamic, parameterized FAQ as opposed to the traditional static FAQ. A question template is a question with entity slots – free space for data instances that represent the main concepts of the question. For example, "When does <performer> perform in <place>?" is a question template where <performer> and <place> are the entity slots. If we fill these slots with data instances that belong to the concepts, we get an ordinary question, e.g., "When does Depeche Mode perform in Globen?"

The question template's "answer" is created by the help of a *database query template* – a formal database query having entity slots for data instances, primarily primary keys. After the slots are filled, the template becomes an ordinary executable database query. The following could be a query template associated with the above question template:

```
SELECT t.time FROM timetable AS t WHERE t.eventid IN
  (SELECT e.eventid FROM event AS e WHERE
   e.placeid = <place> AND e.eventid IN
    (SELECT p.eventid FROM performer AS p WHERE
     p.performerid = <performer>))
```

Processing of a query template and executing the query returns raw data which needs to be formatted and complemented with wrapping text such as a header and footer. The wrapping text complemented with entity slots forms an *answer template*, e.g., "<performer> performs in <place> at time".

We can view a question template as a logical statement with a predicate having variable and fixed parameters, where the fixed parameters are entity slots:

$$\exists\ \text{variable}_1, \text{variable}_2, \ldots \text{variable}_n:$$
$$Q\ (\text{fixed}_1, \text{fixed}_2, \ldots \text{fixed}_m, \text{variable}_1, \text{variable}_2, \ldots \text{variable}_n)$$

If the question has an answer, the above logical statement is true. A database query template embodies the predicate *Q*, it implements the relationships between the parameters. During the question answering process, the values of the fixed parameters – fixed_1, fixed_2, … fixed_m – are bound to the user question, whereas the values of the variable parameters – variable_1, variable_2, … variable_n – are to be found. In order to answer the user question, the system finds all combinations of the variable parameters that retain the value of *Q* true.

Question and answer templates correspond to FAQs and their answers. A new concept, nonexistent in automated FAQ answering, is database query template which implements the relationships between the fixed and variable parameters. The templates are created manually with assistance of computerized tools (see [3] about creating an FAQ entry whose structure is similar to that of a question template) because this process requires analysis of the knowledge domain and understanding the structure of the underlying database. Today's technology does not allow automatic analysis of an arbitrary knowledge domain.

Answering a user question takes the following steps. The question assistant:

1. retrieves data instances that are relevant to the user question (see Section 4);
2. retrieves question templates that match the user question (the FAQ retrieval technique [3] is used);
3. combines the retrieved data instances and question templates, and creates one or several interpretations of the original question; the user selects a desired interpretation, and the question assistant answers it.

3 Question Templates and the Conceptual Model of the Database

The similarity between question templates and FAQs suggests that the mission of the question templates is to embody typical questions. Unlike the FAQs, however, the question templates are bound to the underlying structured, e.g. relational, database.

Conceptual modeling [4] is an activity that involves eliciting concepts, their attributes, relationships, and restrictions from a fuzzy knowledge domain. During the conceptualization process, information is transformed into sentences, sentences into elementary sentences, and elementary sentences into object-role pairs. A conceptual model describes which elementary sentences may enter and reside in the information system [5]. Entity slots in question templates are bound to the concepts, or entities, in the conceptual model while the templates themselves express the relationships between the concepts – those elementary sentences. Fig. 1 shows an example of binding a question template to a piece of a conceptual model.

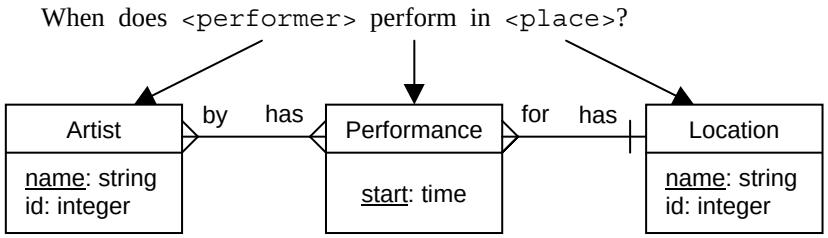

Fig. 1. Question template bound to a piece of a conceptual model

One question template serves a large number of data instances that pertain to its entity slots. If, however, we want to cover a new relationship or attribute, we have to create a new template. The relationships expressed in a question template are static unless we transform them into meta-concepts and treat them as instances in the corresponding entity slots.

During the question-answering process, the question assistant does *not* use the graphical representation of the conceptual model. Instead, it uses an embodiment of the conceptual model in a collection of question templates. The graphical representa-

tion helps when the question templates are created. Whatever information source is used in order to create the templates, they may not contradict the conceptual model as long as such a model is possible.

In conceptual modeling, ISA ("is a") and PartOf are conventional relationships between concepts. The question assistant perceives these relationships as synonymy and context sensitive synonymy.

Establishing synonymy between several entities is an easy task – we define and use a parent entity. Context sensitive synonymy is slightly more complicated. Let us consider the question "When does Depeche Mode perform in Globen?" Because Globen is located in Stockholm (i.e., Globen is a part of Stockholm) and the band arrives from abroad, a more common question would be "When does Depeche Mode come to Stockholm?" In this context Globen can be exchanged with Stockholm, in some others cannot. "How many seats are there in Globen?" is a reasonable question, whereas "How many seats are there in Stockholm?" is not.

There are two options how to handle context sensitive synonymy. First, we can introduce separate equivalent question templates for both entities involved in the PartOf relationship in the context where the synonymy is appropriate. Second, we can define a parent entity and use it only in the context where the synonymy is appropriate.

4 Retrieval of Relevant Data Instances

Retrieval of data instances has three phases: query expansion, retrieval of candidate data instances, and examination of the candidate data instances.

The simplest *query expansion* is stemming of words, which allows matching inflections of the words. The question assistant uses a stem dictionary, which enables better control over the stems and works equally well with English and non-English words, as well as exotic proper names. The stem dictionary is extended by the repository of irregular forms of words and basic synonyms.

After the user query has been expanded, the question assistant selects a number of *data instances that are candidates* for closer examination. A candidate data instance contains at least one word represented in the expanded user query. The data index, which is analogous inverted index or inverted file in Information Retrieval, facilitates the process of selecting the candidates. In the index, a data instance is identified by its primary key and its entity name. Entity names bind the data instances to the entity slots in the question templates, database query templates, and answer templates. The entity names are linked to the physical data carriers – data tables and columns.

The question assistant *examines the candidate data instances* and concludes whether or not a given data instance is relevant to the user query. In the examination process, the question assistant uses entity-specific and data instance-specific rules. *Entity-specific rules* make use of the meaning of the data instance. For example, when the question assistant matches "John Smith" or "Museum of Modern Arts in Stockholm" to the user question, it knows that first is a person name and second is a place name. Such knowledge is helpful dealing with first and last names, prepositions, numbers, proper names, etc. Unfortunately, general rules tend to have exceptions.

Entity-specific rules cannot cope with particular synonyms such as pseudonyms. A *data instance-specific rule* is a list of phrases attached to a particular data instance. Wherever a data instance-specific rule is defined, it overrides the entity-specific rule.

5 Conclusions

The main contribution of this research is adapting the FAQ answering technique [3] to question answering using data in a structured, e.g. relational, database. When the question-answering system, called question assistant, receives a user question, it matches the question to a number of question templates – dynamic, parameterized "frequently asked questions". Unlike a static FAQ, a question template contains entity slots that are replaced by data instances from the underlying database. The entity slots are bound to the concepts, or entities, in the conceptual model of the database while the templates themselves express the relationships between these concepts in form of natural language sentences.

The main advantage of the question assistant is its simplicity. Being a pattern matching system, it requires no sophisticated processing of user questions. Maintenance of the system does not require rare human sills: the maintainer must have sufficient knowledge of the subject domain, a good command of English, and the ability to write database queries. The main disadvantages are limited sensitivity to the user input and generating interpretations as an intermediate step in the question-answering process.

The question assistant, like any existing natural language interface, is *not* recommended for building the only user interface of a database. The system is recommended in situations where answering of typical questions is appropriate, where the conventional keyword-based search retrieves too much irrelevant information.

References

1. Androutsopoulos, I., Ritchie, G. D., Thanisch, P.: Natural Language Interfaces to Databases – An Introduction. In: Journal of Natural Language Engineering, Vol. 1, No. 1, Cambridge University Press (1995)
2. Sneiders, E.: Automated Question Answering: Template-Based Approach. Ph.D. thesis. Department of Computer and Systems Sciences, Stockholm University and the Royal Institute of Technology, Sweden (2002)
3. Sneiders, E.: Automated FAQ Answering: Continued Experience with Shallow Language Understanding. In: Question Answering Systems. Papers from the 1999 AAAI Fall Symposium, November 5-7, North Falmouth, Massachusetts, USA. Technical Report FS-99-02. AAAI Press (1999) 97-107
4. Loucopoulos, P., Zicari, R. (eds.): Conceptual modeling, databases, and CASE: an integrated view of information systems development. John Wiley & Sons (1992)
5. Nijssen, G. M: Current Issues in Conceptual Schema Concepts. In: Nijssen, G. M (ed.) Architecture and Models in Data Base Management Systems. Proceedings of the IFIP Working Conference on Modelling in Data Base Management Systems, Nice, France, 3-7 January 1977. North-Holland Publishing Company (1977) 31-65