EntityManager: An Entity-Based Dirty Data Management System

Hongzhi Wang, Xueli Liu, Jianzhong Li, Xing Tong, Long Yang, and Yakun Li

Harbin Institute of Technology, China,
{wangzh,lijzh,liyakun}@hit.edu.cn, {shally78952286,newyanglong}@163.com,
xiaohuo_2008@yahoo.cn

Abstract. Dirty data exist in many systems. Efficient and effective management of dirty data is in demand. Since data cleaning may result in the the loss of useful data and new dirty data, we attempt to manage dirty data without cleaning and retrieve query result according to the quality requirement of users. Since entity is the unit for understanding objects in the world and many dirty data are led by different descriptions of the same real-world entity, we propose EntityManager, a dirty data management system with entity as the basic unit and keep conflicts in data as uncertain attributes. Even though the query language is SQL , the query in our system has different semantics on dirty data. In the demonstration, we will show a new philosophy for managing dirty data around entities. We will present our prototype allowing load dirty data and query dirty data according to the requirement of users.

1 Introduction

In many systems, dirty data exist because of many reasons. Dirty data will do harm to the applications. Currently, the major method to deal with dirty data is data cleaning. Even though data cleaning could handle dirty data in many cases, such methods have the shortcoming that the repairing in data cleaning may lead to new dirty data and the deletion during data cleaning will result in the loss of data. Without extra information, it is difficult to discover true value of data. Therefore, we attempt to keep dirty data and perform query on dirty data to obtain relative clean results.

Since entity is the basic unit for understanding objects in real world, our idea is to organize tuples according to referred real-world entities. Keeping dirty data, it may occur that an attribute of an entity may have multiple values. Such data could be considered as uncertain data. Even though many uncertain data management systems have been proposed, they are not designed to manage uncertain data generated from entity resolution. They are based on the concept of "possible world", which is difficult to define on dirty data.

Without using the concept of possible world, by managing the data with entity as the basic unit, we develop EntityManager. In our system, entity resolution is performed on the data sets and the tuples corresponding to the same real-world entity are merged as an uncertain tuple which stores different values of an attribute as an uncertain attribute in the relation.

© Springer-Verlag Berlin Heidelberg 2013

With the consideration of the uncertainty in the attributes and possible errors in the constraint in the query, the queries in our system have different semantics with traditional databases. For the ease of usage, EntityMangager accepts common SQL statements and returns results with possibilities representing the degree that this entity matching the query. With such possibilities, users could judge whether the results should be accepted.

In this demo, the audience has the ability to interact with the system through a graphical interface that allows them to load dirty data, input SQL statements and view results with the uncertainties. Users may input dirty data with various sizes, dirty degrees, attribute numbers and attribute widths. Users are able to change these parameters and observe the impact on performance.

The remaining parts of this paper are organized as follows. Section 2 provides the data model in our system. The query processing methods are discussed briefly in Section 3. In Section 4, the demo scenario is proposed.

2 Data Model

In this section, we introduce the data model in our system as well as the semantics of the queries. In the data model of our system, the definitions of database and relation are the similar with those in traditional database [1], while the definitions of attribute and tuple are different.

With data model different from traditional relational model, the query in our system has different semantics. With the consideration of multiple values of an attribute in the relation, the constraints in the query should take the uncertainty in attributes and constraint into consideration in two aspects. On one hand, the comparison between the value of attributes should consider the uncertainty in the values. On the other hand, since the values in attributes and constraints may contain errors, the comparison between the attributes and the values in the constraints are approximate. The details of the constraints are shown in [8].

As the queries have different semantics, the definitions of some data operators are redefined in our system. The projection operator is the same, but the selection and join operators are different.

3 Query Processing

The framework of query processing in EntityManager is the same as that of traditional relational databases [2]. With different semantics in the query language, we develop following three new techniques for efficiently query processing on the uncertain databases organized according to entities.

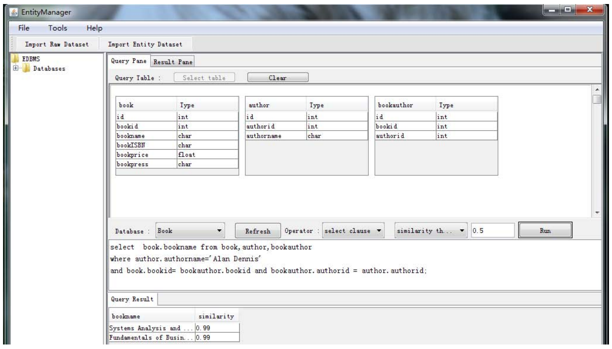
1. *Similarity-based Operators*: With the definitions of selection and join operators different from traditional relational database, we develop similarity search algorithm [6,7] and similarity join algorithm [4] for the entities.

2. *Indices*: To process similarity selection and join efficiently, we designed novel index structures. To handle the special operators in our system, the index considers not only more efficient string similarity search on string attributes [6], but also the similarity search of the combination of numerous and string attributes [7], which can be used for the similarity search on weighted strings.
3. *Query Optimization*: With new operators in our system, for the query optimization, even though the query plan selection algorithms [2] in classical relational databases could be applied in our system, the new estimation techniques for the operators should be developed. Thus we design novel result estimation algorithms for the selection and join operators in [9] and [10], respectively. Additionally, since the selectivity of join on multiple dirty relations is difficult to estimate, we propose a random algorithm for selectivity estimation and join order selection algorithm based the selectivity[5].

4 Demonstration

To demonstrate the features of our system, we load some data of books, paper authors crawled from multiple databases on the web. Since data sources may contain errors or inconsistency, the names or prices of the same book may be different. We apply the entity resolution algorithm in [3] to cluster the relation into entities. For the convenience of users, our system provides the same interface as traditional relational databases including query processing and database maintenance. The uncertainties of the values of attributes are computed by the voting. We attempt to demonstrate our system in following steps.

1. *Data Load*: After the relations in the database are created, the data in format of flat text could be loaded in EntityManager. During data loading, entity resolution is performed and the data is stored in the database in form of entities.



book	Type
id	int
bookid	int
bookname	char
bookISBN	char
bookprice	float
bookpress	char

author	Type
id	int
authorid	int
authorname	char

bookauthor	Type
id	int
bookid	int
authorid	int

```
select  book.bookname from book,author,bookauthor
where author.authorname='Alan Dennis'
and book.bookid= bookauthor.bookid and bookauthor.authorid = author.authorid;
```

bookname	similarity
Systems Analysis and ...	0.99
Fundamentals of Busin...	0.99

Fig. 1. Query Processing Interfaces of EntityManager

2. *Data Browsing* After the data is loaded into the database, user could browse the results of entity resolution with selecting all data in the relation.
3. *Query Proxcessing* As the basic function of a database system, users input the common SQL query and review query results. As shown in Figure 1, the query processing interface is the same as traditional databases. And the query results can be reviewed in detail. To filter the query results according to the probabilities, users could input the threshold or the number of results with the highest possibilities to be selected.

Acknowledgement. This paper was partially supported by NGFR 973 grant 2012CB316200, NSFC grant 61003046, 61111130189 and NGFR 863 grant 2012AA011004. Doctoral Fund of Ministry of Education of China (No.20102302120054). IBM UR Joint Project (No.MH20110819). Key Laboratory of Data Engineering and Knowledge Engineering (Renmin University of China), Ministry of Education (No.KF2011003).

References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley (1995)
2. Garcia-Molina, H., Ullman, J.D., Widom, J.: Database System Implementation. Prentice-Hall (2000)
3. Li, Y., Wang, H., Gao, H.: Efficient entity resolution based on sequence rules. In: Shen, G., Huang, X. (eds.) CSIE 2011, Part I. CCIS, vol. 152, pp. 381–388. Springer, Heidelberg (2011)
4. Liu, X., Wang, H., Li, J., Gao, H.: Es-join: Similarity join algorithm based on entity. Research Report HITDB-12-001, Harbin Institute of Technology (October 2012)
5. Liu, X., Wang, H., Li, J., Gao, H.: Multi-similarity join order selection in entity database. Journal of Frontiers of Computer Science and Technology 6(10), 865 (2012)
6. Tong, X., Wang, H.: Fgram-tree: An index structure based on feature grams for string approximate search. In: Gao, H., Lim, L., Wang, W., Li, C., Chen, L. (eds.) WAIM 2012. LNCS, vol. 7418, pp. 241–253. Springer, Heidelberg (2012)
7. Tong, X., Wang, H., Li, J., Gao, H.: A top-k query algorithm for weighted string based on the tree structure index. In: National Database Conference of China (2012)
8. Wang, H., Li, J., Wang, J., Gao, H.: Dirty data management in cloud database. In: Grid and Cloud Database Management, pp. 133–150 (2011)
9. Zhang, Y., Yang, L., Wang, H.: Range query estimation for dirty data management system. In: Gao, H., Lim, L., Wang, W., Li, C., Chen, L. (eds.) WAIM 2012. LNCS, vol. 7418, pp. 152–164. Springer, Heidelberg (2012)
10. Zhang, Y., Yang, L., Wang, H.: Similarity join size estimation with threshold for dirty data. Journal of Computers 35(10), 2159–2168 (2012)