Warehouse Creation—A Potential Roadblock to Data Warehousing

Jaideep Srivastava, *Senior Member*, *IEEE,* and Ping-Yao Chen, *Student Member*, *IEEE*

Abstract—Data warehousing is gaining in popularity as organizations realize the benefits of being able to perform sophisticated analyses of their data. Recent years have seen the introduction of a number of data-warehousing engines, from both established database vendors as well as new players. The engines themselves are relatively easy to use and come with a good set of end-user tools. However, there is one key stumbling block to the rapid development of data warehouses, namely that of *warehouse population.* Specifically, problems arise in populating a warehouse with existing data since it has various types of heterogeneity. Given the lack of good tools, this task has generally been performed by various system integrators, e.g., software consulting organizations which have developed in-house tools and processes for the task. The general conclusion is that the task has proven to be labor-intensive, error-prone, and generally frustrating, leading a number of warehousing projects to be abandoned mid-way through development. However, the picture is not as grim as it appears. The problems that are being encountered in warehouse creation are very similar to those encountered in data integration, and they have been studied for about two decades. However, not all problems relevant to warehouse creation have been solved, and a number of research issues remain. The principal goal of this paper is to identify the common issues in data integration and data-warehouse creation. We hope this will lead: 1) developers of warehouse creation tools to examine and, where appropriate, incorporate the techniques developed for data integration, and 2) researchers in both the data integration and the data warehousing communities to address the open research issues in this important area.

Index Terms—Data warehouse, entity identification, attribute value conflict, data mining, data integration.

1 Introduction

The ever reducing cost and increasing speed of communication networks has made it possible for organizations to interconnect their information systems, making it possible to use enterprise-wide information for tactical and strategic decision-making. Additionally, the ever increasing capacity and decreasing price of storage devices has made it possible to have historical data on-line, making on-the-fly temporally oriented analysis possible. For the new *enterprise-wide information architectures* now emerging [42], the *data warehouse* (DW) is a critical element. At this point there is a great interest in data warehousing in the industry [23] as well as in academia [45].

The term *data warehouse* was first introduced in [26] and used to describe *a subject oriented, integrated, nonvolatile, and time variant collection of data, in support of management's decision-making.* Classical data management systems are organized around the functions in an organization, e.g., for an insurance company the functions might be auto, health, life, and casualty insurance, while the appropriate subject areas for data organization might be customer, policy, premium, and claim. Heterogeneity of data from different operational systems [30] is a particularly difficult problem faced in creating a data warehouse. The different decisions made by application designers over the years show up in a number of ways, e.g., no consistency in encoding, incompatible naming conventions, different physical attributes and their measurements, etc. Overall, it is essentially a problem of integrating the mental models of the designers, and since the assumptions are hardly ever spelt out, in general this may require extracting the model from the data [11], [18].

A number of commercial (relational) DBMS vendors were quick to realize that optimizing the design of their engines needed to be different for supporting data warehouses, compared to those for *on-line transaction processing* (OLTP) or *decision support* (DSS). Specific optimizations include handling very large database sizes, support for multiple indices, temporally oriented analysis, etc., while taking advantage of not having to provide transaction support. This led a number of relational vendors, e.g., Oracle and IBM, to offer versions of their relational engines optimized appropriately. Some more recent entrants among the relational vendors targeted their engine architecture specifically to address the needs of data warehousing, e.g., Redbrick. Most of the other relational vendors are following suite.

As a followup to the pioneering work in defining a data warehouse [26], Inmon and Hackathorn [25] described the various ways in which a data warehouse may be used. The basic approach is to analyze a wide variety of database applications in large organizations and determine the common usage patterns. These are then grouped according to the *manager's* and *end-user's* perspectives. Once these usage paradigms are formulated, a number of vendors have started offering tools for analyzing the data in the warehouse based on these paradigms, e.g., DecisionSuite from Information Advantage [23], DSS from MicroStrategy [33],

• *J. Srivastava and P.-Y. Chen are with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455. E-mail: {srivasta, pchen}@cs.umn.edu.*

Manuscript received 12 June 1997; revised 30 Oct. 1998.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 108361.

1041-4347/99/$10.00 © 1999 IEEE

and SQL-server 7.0 from Microsoft [32]. Analysis paradigms like *on-line analytical processing* (OLAP) [6] have been developed, which support the operations of *rolling-up* and *drilling-down* in addition to standard SQL querying. Rolling-up is the process of going from a set of data elements to some aggregation defined on it, where the original set of data elements themselves may be aggregations on some lower level data sets. Drilling-down is the exact opposite of rolling-up, i.e., going from an aggregation to its constituent data elements. Recent research has proposed extending the relational algebra with a lattice-shaped data structure called a *data-cube* and operations upon it [15]. The research community has started addressing algorithmic and optimization issues related to the data-cube operator [21].

One of the outcomes of the *business process re-engineering* wave of the late 1980s has been the realization by large organizations that their functioning can be made much more effective by developing a systematic enterprise-wide data management architecture, e.g., [26], [43]. Section 2 provides the details of such an architecture. A key element of this architecture is the data warehouse which forms the foundation for new kinds of enterprise-wide analyses of information. A number of actual experiences with building data warehouses, e.g., Cargill Inc. [13] and United Health Care [3], have shown that a task critical to the effective use of warehousing is the creation of the warehouse by extracting data from existing legacy systems and using it to populate the warehouse. Specifically, pilot projects carried out by these organizations showed that a lot of knowledge-intensive human effort was required for the task. consulting services that address this issue. A recent study [19] assesses the situation with data warehouse creation tools as *unclear whether using these tools will reduce effort or increase it*.

Based on the above, we believe that there is a big unmet need for effective and easy to use processes and tools to help the task of data warehouse creation and population. This can seriously hamper the widespread adoption of data warehousing in an enterprise-wide manner, and thus prevent organizations from taking full advantage of them. On the other hand, there has been a substantial amount of research in the area of data integration, under the title of *federated databases* and *heterogeneous distributed databases* [40]. Many of the problems addressed in this research are exactly the ones encountered in the task of warehouse creation. The principal contribution of this paper is to show the commonality of issues in data integration and warehouse creation, provide a brief survey of the relevant results, and identify open research areas. Thus, we hope to provide the builders of warehouse creation tools source of potential ideas, and researchers in data warehousing a set of interesting research problems.

This paper is structured as follows: In Section 2, we present a generic architecture of enterprise data management, and show how data warehousing fits into it. In Section 3, we discuss the issues in warehouse creation while, in Section 4, we provide a survey of the work in this area. Section 5 discusses open research issues, and Section 6 concludes the paper.

2 Enterprise Data Management Architecture

Fig. 1 shows a fairly standard modern enterprise data management architecture for a large organization, with *tactical*, i.e., day-to-day, as well as *strategic*, i.e., long term, data management and analysis needs. It consists of operational data management systems which support existing applications by managing current data. It also consists of a *corporate data warehouse* and a number of *data marts* on which various kinds of strategic analyses are performed. In the following, we briefly describe the various elements of the architecture.

Operational Data Systems: These are traditional relational and other data systems, which are used for the day-to-day operations of the organization. They contain up-to-date data, and provide interactive access, both for transaction and decision-support systems. The database size can range from tens of megabytes to a few gigabytes, which though large in itself is quite small compared to the size of the corresponding warehouse. Since these databases are regularly updated, usually by concurrently executing applications, a critical goal for them is to maintain transaction consistency. In addition, since many of the applications are mission critical, e.g., airlines and banking, there is usually a need to mask system failures from the application. Finally, due to interactive access, response time and throughput requirements are stringent.

Warehouse Creation: For a warehouse to be used effectively, it is important to pay sufficient attention to its creation. As shown in Fig. 1 this includes *architecture selection*, *warehouse schema creation*, and *warehouse population*. These issues are discussed in detail in subsequent sections.

Corporate Data Warehouse: This is the principal repository of historical information in the organization, and stores data instances for the enterprise data model. Data is entered into this repository periodically, usually in an append-only manner. Sometimes the data here may directly be used for analysis. However, in most cases focused sets of data are extracted into smaller *data marts* where they are analyzed [26]. Since this is often the definitive data in the organization, in many instances such a warehouse has also been called the *foundation data*. Since data is usually entered or extracted in a batch mode, interactive response is not a significant issue.

Data Marts: Data marts are smaller data warehouses, usually focusing on a small subset of the enterprise data. Typically each mart is used by a particular unit of the organization for various strategic analyses relevant to its goals. Data is extracted from the corporate warehouse into the data mart periodically, and used for analysis. Interactive response is an issue in data marts as interactive analysis tools work directly on the data.

Warehouse Analysis Tools: A number of tools have been built for performing strategic analysis of warehouse data. As shown in Fig. 1, these include trend analysis, data mining, simulation, forecasting, and on-line analytical processing.

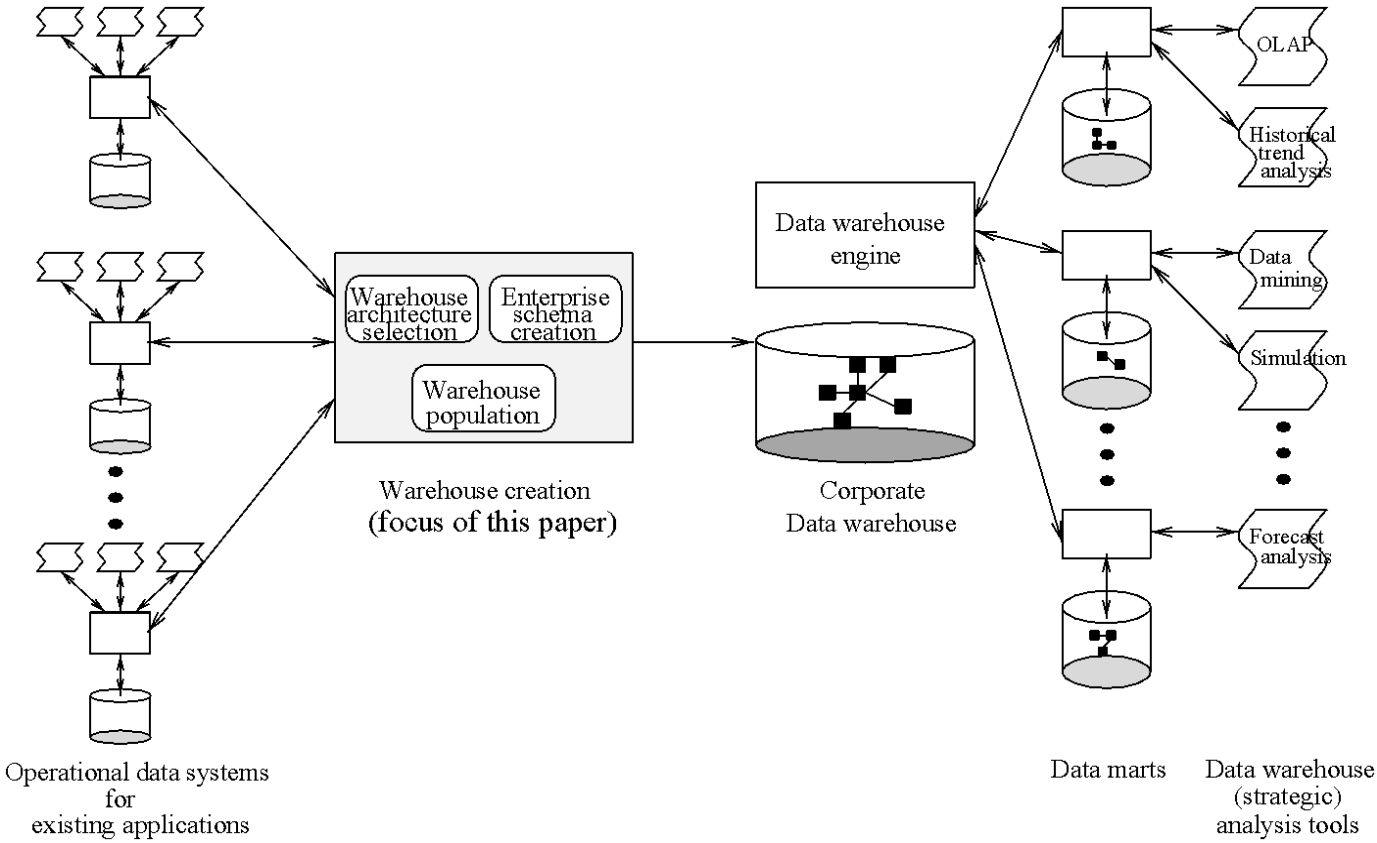

Fig. 1. Enterprise data management architecture.

3 Issues in Data Warehouse Creation

A number of issues must be addressed in building a data warehouse. Some are associated with the creation of the warehouse and others with its operation. In the following, we discuss the issues affecting warehouse creation.

3.1 Warehouse Architecture Selection

A number of architectural approaches to data warehousing are possible, and selection between them is affected by factors such as size, nature of use, etc.

Database Conversion: Shown in Fig. 2, this architectural choice involves taking all the data in the source systems and performing a (often one-time) conversion to the (integrated) target system. This approach is applied when:

1) the source systems are to be retired and all the data is being moved to the target system, and
2) both the source and target systems will continue to be in operation, and this is the initial population of the target database.

The data in the source systems is mapped into a global model and then written out/exported to the target database. In this approach, bulk data conversion is being performed, and hence optimization of the processing is important. Integrating data from multiple sources can require comparison between all pairs of data items, i.e., the cost can in general be quadratic with respect to the input size. Since input data can typically be in the order of tens of millions or higher [27], this optimization is quite critical.

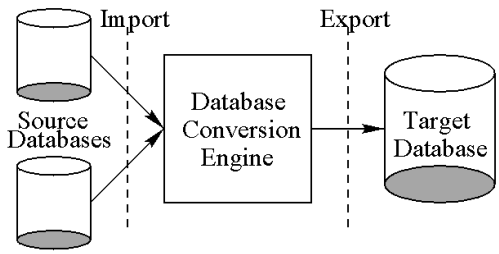

Fig. 2. Database conversion.

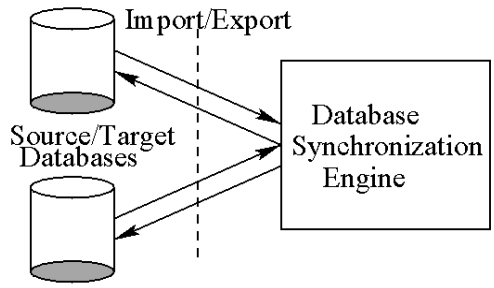

Fig. 3. Database synchronization.

Database Synchronization: Fig. 3 shows the synchronization architecture. In this case, there is an existing warehouse, and data is extracted periodically from the operational source systems to update the target system, thus *synchronizing* the target system to the source systems.

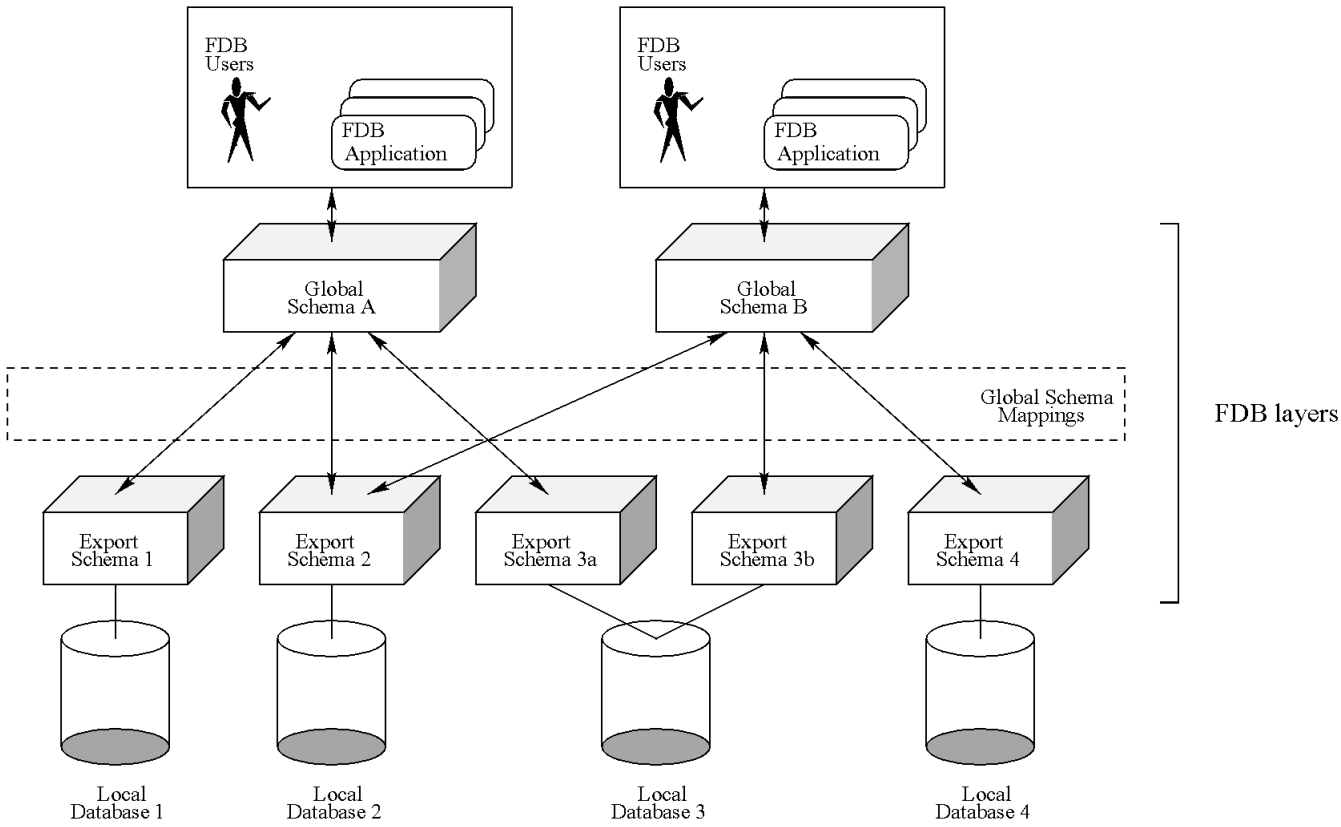

Fig. 4. Federated database.

The key differences between this architecture and the previous one are:

1) the volume of data handled is much smaller, and
2) information being produced for the target database already has its counterpart from before, with which it may require integration.

The main issue is one of using incremental algorithms to optimize the processing. A lot of recent work in materialized view maintenance [16] is relevant to this problem.

Federated Database: In some cases the number of users of the enterprise-wide integrated data are few, and having a separate full-fledged data warehouse may be an overkill. In addition, there may sometimes be a requirement for having the target database be completely up-to-date. In such cases, a *federated database* (FDB) may be the best architectural choice, as shown in Fig. 4.

In this architecture, only the data required by a query is integrated. Hence, the optimization focuses on handling small amounts of data as well as fast interactive response time. Since query processing and data integration must be handled together, the optimizer must consider both simultaneously [29].

3.2 Enterprise Schema Creation

With any of the data warehouse architectural choices discussed above, the first step is to develop an enterprise schema [26]. The enterprise schema is the output of *enterprise data modeling* [26], and captures the main entities about which the enterprise maintains information and relationships between them. It is invariably the case that various units of an organization have existing schemas for their respective portions of the enterprise-wide information [40]. These are used as the starting point for the enterprise schema. Thus, an important task for enterprise schema creation is to integrate various existing schemas into a global one, refining it, and filling out any missing elements. In addition, data warehouses provide explicit support for the historical dimension as well as aggregation, which must be modeled as special entities. Finally, the union of the enterprise data may give rise to new entities which must be modeled in the schema.

Enterprise Data Model: A data model for the enterprise must be first developed, to identify the key entities in the organization and their relationships to each other [26]. In an ideal situation, such modeling would begin by analyzing the various processes in the organization and extracting the relevant entities and relationships from them. However, in most situations it is much more realistic to use the existing entities and relationships, modeled by the schemas of existing components data sources, as a starting point. Analysis of various organizational processes helps in refining the enterprise data model. Experience has shown that there is no *perfect* enterprise data model, and hence this task can never be 100 percent complete. Thus, once a reasonable model has been obtained it must be put into operation, with provision for subsequent modifications.

Schema Integration: In creating the enterprise data model, an all too often encountered situation is one where multiple, independently developed databases model different (and sometimes overlapping) aspects of the same set of real-world entities. However, having been developed independently, the schemas do not match well. *Structural heterogeneity* is one class of discrepancies which arise, e.g., different field sizes, units, or scales. *Semantic heterogeneity* is another class of discrepancies, which includes the *synonym* and the *homonym* problems. An example of the former is the names *EMP* and *EMPLOYEE* being used to refer to the same class of *real-world employees* in two different databases. An example of the latter is the name *EMP* being used to refer to *real-world employees* in one database, and to real-world employers in another. The reason for these discrepancies is that the designers of the respective databases were targeting different sets of applications, even though for the same set of real-world entities, and hence captured different (but overlapping) models of reality in their databases. Hence, overall the *schema integration* problem is one of *integrating multiple overlapping models of a set of real-world entities*. Specific technical approaches include automatic and semiautomatic ways to determine synonyms, antonyms, IS-A relationships, etc. A large body of related work exists in the database and knowledge representation community. Batini et al. [4] and Sheth and Larson [40] provide a good survey of the database work in this area.

Constraints: In addition to the structural and semantic mismatches of schema entities, there is the additional problem of constraint mismatches, which are often not evident from the definitions of entity types. For example, one database may have a constraint such as *EMP.age <= 60* while another may have the constraint on age as *EMP.age <= 65*. In integrating such databases there seems to be in general no *right approach* to resolving such constraint incompatibilities [38]. For example, we can take the approach that the new constraint should be *EMP.age <= 65* since it is the weaker of the two, and hence all data is guaranteed to satisfy it. However, tightness of constraints in a schema is usually an indication of the quality of the data in the database, and such an approach eventually degrades the overall quality of the data to its lowest common denominator. An alternative approach is to select some tighter constraint at integration time, based on existing constraints in participating databases, and perform the requisite *data quality improvement* through the use of tools and human intervention, to ensure that the integrated data has at least this data quality. Using an object model for the integrated database, another possible approach is to define two employee sets, namely one with *EMP.age <= 60* and the other with *EMP.age <= 65*, with an IS-A relationship between the two sets. There has been some initial work in addressing these issues [17], [38].

3.3 Warehouse Population

Once the component schemas have been integrated to develop the global schema of the warehouse, with acceptable resolution of schema mismatches, the next step is to populate the warehouse with data. Experience has shown that even though the schemas may have been integrated, there may still be problems in integrating the specific data instances [10], [28], [30], [26].

Semantic Issues: A number of semantic discrepancies arise in integrating data instances from multiple sources. A specific problem is *entity identification* [30], namely determining whether a pair of records (coming from two different databases) represent the same or different real-world entities. This becomes a difficult problem because usually there is no common key which can be used as an identifier for the union of the two sets of records. Another is the *attribute value conflict resolution* problem. This occurs when records from different databases have been matched, i.e., *entity identification* has occurred, but it is found that values of the same attribute for an entity instance, coming from different data sources, are different. There can be a number of possible sources of such errors, e.g., data-entry errors, different policies of database maintenance, and modeling of slightly different concepts. An example of different maintenance policies is that in one database the employee age is updated every January 1, while the other does so every July 1, which can lead to a discrepancy of one year in the age of some employees between January 1 and June 30. An example of modeling slightly different concepts is having *salary* in one database and *income* in the other. As data warehouses are being built, experience with such problems are being generated, and various ad hoc solutions have been proposed. A systematic study of semantic heterogeneity issues in warehouse population has only just begun [29], [31].

Scalability: Since warehouses store information about the database as it progresses over time [26], they tend to grow much more rapidly than on-line databases. It is quite common to start with an initial warehouse of size 10-100 gigabytes, and subsequently have a periodic (weekly or monthly) update rate of 1-10 gigabytes. The processing described above for warehouse population must be carried out for the initial population as well as for every periodic update. The data integration tasks typically range in complexity from $O(nlgn)$ to $O(n^2)$ for n data items [22]. Given tens to hundreds of gigabytes of data, this can be very time consuming, and hence there is a need for improved algorithms for data integration tasks. Furthermore, since these tasks are heavily set-oriented, data-parallel computing techniques appear promising, and should be investigated.

Incremental Updates: As data is added to an existing warehouse, it must be integrated with pre-existing data. The success of incremental update techniques in on-line databases must be extended to this type of processing. Examples of such techniques include incremental algorithms and indices. As of now hardly any work has been done in this area.

4 Survey of Literature Relevant to Warehouse Creation

In this section, we provide a brief review of the literature relevant to warehouse creation. Since the main goal of this paper is to show the commonality between the issues in warehouse creation and data integration, the focus of this survey is on the problems proposed in the research domain, and their solutions. Thus, this survey is not comprehensive and does not discuss many important issues. For example, it does not discuss the important issue of tools and process for handling legacy systems developed by a number of system integrators, mostly consulting companies. A good survey of these is provided in chapter 10 of the excellent book by Brodie and Stonebraker [5]. Other good resources include [2], [14], [24], [34], [35]. The specific focus of our survey is on approaches and techniques reported in the literature for

1) warehouse schema creation, including enterprise data modeling, schema integration, and constraint integration,
2) warehouse population, including entity identification, attribute value conflict resolution, and scalability issues, and
3) mining knowledge for warehouse creation.

4.1 Schema Creation

Enterprise Data Modeling: Developing an enterprise data model is the first step in building a data warehouse. Typically this is not very different from standard database design except for handling schema mismatches, which is discussed next. This is a well studied area in the database modeling and design literature, e.g., [12], and we do not discuss it any further.

Schema Integration: The survey paper of Batini et al. [4] lists the steps and goals of the schema integration process, and compares a dozen methodologies for schema integration. It divides schema integration activities into four steps, namely preintegration, comparison, conformation, and merging and restructuring. It provides a comprehensive survey of the pre-1986 schema integration literature according to the proposed taxonomy. Three tools developed to perform schema integration are reported in Hayes and Ram [20], Sheth et al. [39], and Souza [41]. Sheth and Larson [40] provides a more up-to-date survey of the field. While schema integration is by no means a completely solved problem, there has been relatively less activity in the area in the past few years. The principle conclusions are that

1) it is likely to remain a largely human intensive task, and
2) good tools can help reduce the tedium from human being, allowing him to focus on the more conceptually difficult parts.

Constraint Integration: The problem of constraint mismatch during data integration was first introduced in [17], concluding that in general attempts to enforce global constraints were futile. A subsequent proposal [38] came up with the idea that we must draw upon some domain and application semantics to develop policies for integrating database constraints. We believe this to be a very promising idea, as it proposes a natural way out of a difficult situation; especially since emulates what is done in practice under the name of *business rules* or *operational procedures*. Unfortunately, there has not been much follow-up on this idea. We believe this area will become popular with the growing importance of data warehouses, especially since constraint enforcement is a good way to ensure data quality.

4.2 Warehouse Population

Entity Identification: Kent [28] first used the term "the breakdown of the information model" to indicate that information about a real-world entity may be modeled differently in independently developed databases, with no obvious way to correlate them. Lim et al. [30] further showed that it is not always possible to integrate data instances even when the schemas are compatible, and therefore new techniques must be developed. They proposed the use of an *extended key*, which is the concatenation of keys (and possibly other attributes) from the relations to be matched, and its corresponding *identity rule* to determine the equivalence between tuples from relations that may not share any common key. Most approaches assume that some *common key* exists amongst relations from different sources, and therefore the key can be used to identify entities. Multibase [9] is an example of such an approach. However, the relations may have no common key, even though they share some common key attributes. The Pegasus project [1] proposed to let users specify equivalence of records across databases. Pu proposed that a portion of the key values be used to match records [36]. Chatterjee and Segev [7] proposed a probabilistic reasoning approach for determining entity equivalence using data associated with common attributes amongst relations. While their method may potentially produce incorrect matching, the use of rules to resolve this problem has also been suggested by Wang and Madnick in [46]. The rule based approach introduces additional semantics to the solution, yet due to the heuristic nature of knowledge, it is possible to produce inaccurate results. Hernandez et al. [22] identify this problem as the Merge/Purge problem and propose multiattribute key equivalence as the primary matching mechanism. Due to its importance, this problem has received wide attention from the industry, e.g., [2], [14], [24], [34], [35]. However, an overall systematic approach does not exist.

Attribute Value Conflict: Resolving domain incompatibility among independently developed databases often involves uncertain information. DeMichiel [10] shows that uncertain information can be generated by the mapping of conflicting attributes to a common domain, based on some domain knowledge. She proposed the use of partial values to represent uncertain information from source databases. Lim et al. [31] show that uncertain information can also arise when the database integration process requires information not directly represented in the component databases, but is obtained through some summarization of data. Several approaches to the attribute value conflict problem

have been proposed in the past: Dayal [8], [9] proposed the use of aggregate functions such as average, maximum, minimum, etc. to resolve discrepancies in attribute values. Tseng et al. [44] proposed the notion of partial values to capture uncertainty in attribute values. Lim et al. [31] proposed an extended relational model based on Dempster-Shafer theory of evidence to incorporate the uncertain knowledge about the source database. However, not much actual experience with these approaches exists.

4.3 Mining Knowledge for Warehouse Creation

It is widely accepted that data integration is a knowledge intensive task. Some proposed approaches have assumed that such knowledge is available from human sources. Some experiences in warehouse creation [37] have shown that in this domain it is almost impossible to find people who know enough about the various data sources to act as experts. In most cases, however, enough system users exist who can provide examples of what is correct and what is incorrect. This makes *data mining* an appropriate technology to use for extracting integration knowledge from the data. Dao and Perry [11] was the first to introduce data mining techniques to the problem of data integration, using it for schema integration. Ganesh et al. [18] showed the use of data mining techniques to learn knowledge for instance integration tasks like entity identification and attribute value conflict resolution.

5 Research Issues

In the previous sections, we discussed various issues in data warehouse creation and provided a survey of the related literature. In this section we outline some of the problems facing the builders of data warehouses, and formulate them in terms of open research problems.

5.1 Schema Integration

More than two decades of schema integration research has yielded a number of techniques, some of which have been incorporated into commercial and/or public domain tools. The techniques used in these tools are often ad hoc, working well in special cases but are often not scalable to other applications. An important consensus in the research community is that schema integration is a complex task and it does not seem possible to be completely automate it. Addressing the following research issues would make significant practical and theoretical advances in this area:

1) Modeling schema integration as a task in model integration, with a solid logic-based foundation. This would help in capturing the precise information available, and enable the use of logical inference for resolving conflicts and mismatches.
2) Development of an interactive schema integration tool, perhaps with a logic-based back end, that works with schema designers/integrators to develop the integrated schema.

5.2 Instance Integration

Instance integration problems have been recognized and formulated by data base researchers in the last few years [28], [30]. The wide popularity of data warehouses has made them increasingly important, as they continue to become critical bottlenecks in warehouse population [3], [13]. While some interesting approaches have been proposed, we believe a number of important issues remain to be addressed, including the following:

1) The problem of comparing data instances from different source databases to determine if they represent the same real-world entity is a complex one. In its full generality it can be modeled as a problem of *clustering*, where the data instances represent points and real-world entities represent clusters. This formulation, and the applicability of clustering algorithms, needs to be investigated. Since clustering is an inherently approximate process, the trade-offs between clustering accuracy and algorithm performance must be investigated.
2) Integrating data from multiple sources is an inherently approximate (or fuzzy) process. Measures of matching accuracy, and algorithms that take advantage of them, should be developed.
3) The use of domain knowledge should be used in a systematic manner in the process of integration

5.3 Scalability of Integration

As reported earlier, many integration algorithms have quadratic complexity. In many cases, the number of records is too large for such an algorithm. For example, over the last weekend of every month a major financial institution has to solve the integration problem with about 11 million records [27]. They are barely able to meet this deadline. Specific research issues to be addressed include:

1) Algorithmic improvements should be made to not require pairwise comparison between all records.
2) Given the set-oriented nature of much of data integration processing, data parallelization seems to be an attractive approach, and should be investigated in detail.

5.4 Integration Quality

Many actual integration experiences have shown that integration tasks follow a kind of law of diminishing return, i.e., a large portion of the work can be done in a reasonable time; and beyond this the marginal benefit obtained from each unit of effort spent rapidly decreases [37]. Metrics should be developed to quantify the accuracy of integration obtained. Trade-offs between the time spent on integration and the resulting accuracy must be analyzed, and algorithms that can take advantage of this trade-off must be designed.

5.5 Mining for Integration Knowledge

In the previous section, we reviewed literature on the application of data mining to data integration. However, only the surface has been scratched so far, and a number of

issues remain. Following are some of the specific research issues in this area:

1) Data mining techniques, such as supervised and unsupervised learning, should be investigated as suitable candidates for mining integration knowledge. Examples of such knowledge include rules that provide evidence for or against two objects from different databases representing the same real-world entity. Another example would be rules to determine the attribute values in the integrated database.
2) Most mining techniques assume that data is in a fairly clean and complete form, which in general is not true for data to be entered into a warehouse. Thus, there is a need to develop data cleansing techniques that can be used before mining on it is done.

6 Conclusions

In this paper, we make the observation that data warehouse creation is an important task which is increasingly becoming a bottleneck, preventing the rapid deployment of data warehouses [13], [3]. While a number of techniques and software exist for storing the data warehouse, and performing analyses on it, there is a marked lack of tools for the warehouse creation task. Furthermore, it has been observed that doing this on an ad hoc basis has proven to be labor intensive, error prone, and generally frustrating. However, the picture is not as grim as it appears. The problems that are being encountered in the course of warehouse creation have a very high degree of overlap with those encountered in data integration, which have been studied for about two decades. However, not all problems relevant to warehouse creation have been solved, and a number of research issues remain. The principal goal of this paper has been to identify the common issues in data integration and data warehouse creation. We hope this will lead:

1) the developers of warehouse creation tools to examine, and where appropriate incorporate, the techniques developed for data integration, and
2) the researchers in both data integration and data warehousing community to address the open research issues in this important area.

Acknowledgments

The ideas presented in this paper have developed over the years as part of the Myriad federated database project at the University of Minnesota. They have been refined through discussion with a number of people who have been affiliated with Myriad over the years. Specifically, we would like to acknowledge the contributions made by Ee-Peng Lim, San-Yih Hwang, James P. Richardson, Satya Prabhakar, Jian-Dong Huang, Madhaven Ganesh, Jian-Zhong Li, Travis Richardson, Kajal Mediratta, Sharon Yang, Dave Clements, Satish Musukula, and Sujal Parikh.

This work is supported, in part, by the U.S. Department of Transportation through Grant No. USDOT/DTRS93-G-0017 to the University of Minnesota.

References

[1] R. Ahmed, P. DeSmedt, W. Du, B. Kent, M. Ketabchi, W. Litwin, A. Rafii, and M.-C. Shan, "The Pegasus Heterogeneous Multidatabase System," *Computer*, vol. 24, no. 12, pp. 19–27, Dec. 1991.
[2] Apertus Inc., white paper, URL: http://www.apertus.com/ .
[3] J. Bain, "A United Health Care Perspective on Business Information Strategies," *Putting the Data Warehouse on the Internet*, May 1997.
[4] C. Batini, M. Lenzerini, and S.B. Navathe, "A Comparative Analysis of Methodologies for Database Schema Integration," *ACM Computing Surveys*, vol. 18, no. 4, pp. 323-364, Dec. 1986.
[5] M.L. Brodie and M. Stonebraker, *Migrating Legacy Systems: Gateways, Interfaces, and The Incremental Approach*, Morgan Kaufmann, 1996.
[6] E.F. Codd, S.B. Codd, and C.T. Salley, "Providing OLAP (On-line Analytical Processing) to User-Analysts: An IT Mandate, technical report, E.F. Codd and Associates, 1993.
[7] A. Chatterjee and A. Segev, "Data Manipulation in Heterogeneous Databases," *SIGMOD Record*, vol. 20, no. 4, pp. 64–68, ACM, Dec. 1991.
[8] U. Dayal, "Processing Queries Over Generalized Hierarchies in a Multidatabase Systems," *Proc. Ninth Int'l Conf. Very Large Data Bases*, pp. 342–353, Florence, Italy, Oct. 1983.
[9] U. Dayal, "Query Processing in Multidatabase Systems," W. Kim, D.S. Reiner, and D.S. Batory, eds., *Query Processing in Database Systems*, pp. 81–108, Springer-Verlag, 1985.
[10] L.G. DeMichiel, "Resolving Database Incompatibility: An Approach to Performing Relational Operations Over Mismatched Domains," *IEEE Trans. Knowledge and Data Eng.*, vol. 1, no. 4, pp. 485-493, Dec. 1989.
[11] S. Dao and B. Perry, "Applying A Data Miner to Heterogeneous Schema Integration," *Proc. First Int'l Conf. Knowledge Discovery in Databases*, pp. 63-68, Montreal, Canada, Aug. 1995.
[12] R. Elmasri and S.B. Navathe, *Fundamentals of Database Systems*, Benjamin/Cummings, 1993.
[13] C. Faison, "Web Enabled Data Warehouses at Cargill," *Putting the Data Warehouse on the Internet*, May 1997.
[14] Firstlogic Inc., URL: http://www.firstlogic.com/ .
[15] J. Gray, S. Chaudhary, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh, "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub Totals," *Knowledge Discovery and Data Mining*, vol. 1, no. 1, pp. 29-54, Mar. 1997.
[16] A. Gupta, V. Harinarayan, and D. Quass, "Aggregate-Query Processing in Data Warehousing Environments," *Proc. 21st Int'l Conf. Very Large Data Bases*, pp. 358-369, Zurich, Sept. 1995.
[17] H. Garcia-Molina, "Global Consistency Constraints Considered Harmful for Heterogeneous Database Systems," *Proc. First Int'l Workshop Interoperability in Multidatabase Systems*, pp. 248-250, Kyoto, Japan, Apr. 1991.
[18] M. Ganesh, J. Srivastava, and T. Richardson, "Mining Entity-Identification Rules for Database Integration," *Proc. Second Int'l Conf. Knowledge Discovery in Databases*, pp. 291-294, Portland, Ore., Aug. 1996.
[19] J. Hill, "Distinguishing Data Movement Technologies," Gartner Group Research Note, Strategic Data Management, Apr. 1998.
[20] S. Hayes and S. Ram, "Multi-User View Integration System (MUVIS): An Expert System for View Integration," *Proc. Sixth IEEE Int'l Conf. Data Eng.*, pp. 402–409, Los Angeles, Feb. 1990.
[21] V. Harinarayan, A. Rajaraman, and J.D. Ullman, "Implementing The Data Cube Efficiently," *SIGMOD Record*, vol. 25, no. 2, pp. 205–216, ACM, June 1996.
[22] M.A. Hernandez and S.J. Stolfo, "The Merge/Purge Problem for Large Databases," *SIGMOD Record*, vol. 24, no. 2, pp. 127-138, ACM, June 1995.
[23] "OLAP: Scaling to The Masses," Information Advantage White Paper, Minneapolis, Minn., 1997.
[24] Idcentric Inc., "Customer Data Quality, A White Paper," URL: http://www.idcentric.com/ .
[25] W.H. Inmon and R.D. Hackathorn, *Using The Data Warehouse*, John Wiley and Sons, 1994.
[26] W.H. Inmon, *Building the Data Warehouse*, John Wiley and Sons, 1992.
[27] A. Jenks, "Enterprise Data Strategy for Norwest Inc.," private communication, June 1995.

[28] W. Kent, "Breakdown of the Information Model," *SIGMOD Record*, vol. 20, no. 3, pp. 10-15, ACM, Sept. 1991.
[29] E.-P. Lim, J. Srivastava, and S.Y. Hwang, "An Algebraic Framework for Multidatabase Queries," *Distributed and Parallel Databases*, vol. 3, no. 3, pp. 273-307, July 1995.
[30] E.-P. Lim, J. Srivastava, S. Prabhakar, and J.P. Richardson, "Entity Identification in Database Integration," *Proc. Ninth IEEE Int'l Conf. Data Eng.*, pp. 294-301, Vienna, Apr. 1993.
[31] E.-P. Lim, J. Srivastava, and S. Shekhar, "An Evidential Reasoning Approach to Attribute Value Conflict Resolution in Database Integration," *IEEE Trans. Knowledge and Data Eng.*, vol. 8, no. 5, pp. 707-723, Oct. 1996.
[32] Microsoft Corp., "SQL-Server 7.0: Decision Support System," product announcement, Redmond, Wash., 1998.
[33] MicroStrategy, "The Decision Support Systems (DSS)," product announcement, Viena, Va., 1998.
[34] Platinum Inc., URL: http://www.platinum.com/ .
[35] Postalsoft Inc., URL: http://www.postalsoft.com/ .
[36] C. Pu, "Key Equivalence in Heterogeneous Databases," *Proc. First Int'l Workshop Interoperability in Multidatabase Systems*, pp. 314–316, Kyoto, Japan, Apr. 1991.
[37] T. Richardson and J. Srivastava, "Enterprise/Integrator: Using Object Technology for Data Integration," *Proc. Object-Oriented Programming Systems, Languages, and Applications Workshop Object Oriented Integration of Legacy Data Systems*, Austin, Texas, Oct. 1995.
[38] M. Rusinkiewicz, A. Sheth, and G. Karabatis, "Specifying Interdatabase Dependencies in A Multidatabase Environment," *Computer*, vol. 24, no. 12, pp. 46-54, Dec. 1991.
[39] A. Sheth, "Building Federated Database Systems," *Newsletter*, Distributed Processing Technical Committee, vol. 10, no. 2, pp. 50–58, 1988.
[40] A. Sheth and J. Larson, "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases," *Computing Surveys*, vol. 22, no. 3, pp. 183-236, ACM, Sept. 1990.
[41] J. Souza, "SIS: A Schema Integration System," *Proc. Fifth Nat'l Conf. BNCOD*, pp. 167–185, Canterbury, United Kingdom, July 1986.
[42] R. Tanler, "The Intranet Data Warehouse," *Putting the Data Warehouse on The Internet*, May 1997.
[43] L.A. Taylor, "Cargill's Informational Technology Strategy," white paper, Cargill Inc., Oct. 1993.
[44] F.S.-C. Tseng, A.L.P. Chen, and W.-P. Yang, "Answering Heterogeneous Database Queries with Degrees of Uncertainty," *Proc. Second Int'l Conf. Parallel and Distributed Information Systems*, vol. 1, no. 1, pp. 281–302, Jan. 1993.
[45] J. Widom, "Research Problems in Data Warehousing," *Proc. Fourth Int'l Conf. CIKM*, pp. 25–30, Baltimore, Md., Nov. 1995.
[46] Y.R. Wang and S.E. Madnick, "The Inter-Database Instance Identification Problem in Integrating Autonomous Systems," *Proc. Fifth IEEE Int'l Conf. Data Eng.*, pp. 46–55, Los Angeles, Feb. 1989.

Jaideep Srivastava received the BTech degree in computer science from the Indian Institute of Technology, Kanpur, India, in 1983; and the MS and PhD degrees in computer science from the University of California, Berkeley, in 1985 and 1988, respectively. He has been on the faculty of the Department of Computer Science and Engineering of the University of Minnesota, Minneapolis, since 1988, and is currently an associate professor. He served as a research engineer with Uptron Digital Systems in Lucknow, India, in 1983. He has published more than 110 papers in refereed journals and conference proceedings in the areas of databases, parallel processing, artificial intelligence, and multimedia; and he has delivered a number of invited presentations and participated in panel discussions on these topics. His professional activities have included service on various program committees, and he has refereed papers for varied journals and proceedings—including for events sponsored by the National Science Foundation. He is a senior member of the IEEE, and a member of the IEEE Computer Society and the ACM.

Ping-Yao Chen received his BE degree at Tamkang University in Taiwan, Republic of China, in 1990, and his MS degree at the University of Minnesota in 1998. He is currently a PhD student in the Department of Computer Science and Engineering at the University of Minnesota. His research interests include data integration and data mining. He is a student member of the IEEE.