A New System Architecture for Flexible Database Conversion

SITI Z.Z. ABIDIN, SUZANA AHMAD, WAEL M.S. YAFOOZ
Faculty of Computer and Mathematical Sciences,
University Technology MARA,
40450, Shah Alam, Selangor
MALAYSIA
sitizaleha533@salam.uitm.edu.my, suzana@tmsk.uitm.edu.my, waelmohammed@hotmail.com

Abstract: - Much research has been undertaken to work on database sharing, integration, conversion, merging and migration. In particular, database conversion has attracted researchers' attention due to the rapid change in the computer technology. There are also several tools available on the Web for free usage on handling database conversion. All of these works are focusing on the relational database which consists of an integrated collection of logically related records. Database plays an important role in an area of computing when there is a lot of data and information need to be stored and retrieved. Today, databases are used in many disciplines such as business, education, general administration, medicine and many more. Research in database works have been advanced from file management system to data warehousing with the discussion of how a database can be significantly sustainable and have the potential to be added and modified to suit the current situation and technology. In this paper, we propose a new method on doing database conversion by providing a single master database that can accept multiple databases of any type through the use of Java Database Connectivity (JDBC) and application program interface (API). The key contribution of this method is the ability to accept a single record, multiple records or the whole records of a database to be converted into any other database type. Thus, any existing form of database can be integrated and updated without the need to design new database system for coping with the new technology. In this way, the old or existing databases can be used for an unlimited lifetime and a broader scope of application domains.

Key-Words: - Database, Data sharing, Integration, Conversion, Access, Integrity, Migration

1 Introduction

Database systems have become very important in everyday life. Most people do not realize that they are interacting with database systems when dealing with computers. In this information age, many people use computers to store and retrieve data. In networked environments, which provide people with massive amount of information, large storage for keeping information must exist to support such environments. The systems include digital libraries, online bank transactions or internet browsing. According to McFadden *et al.* [1], a database system is an integrated collection of logically related records or files consolidated into a common pool that provides data for one or multiple uses. In addition, a database system is a way of organizing information on a computer that is implemented by a set of computer program.

Database Management System (DBMS) is a set of software that is used to define, store, manipulate and control data in the database. Today, most businesses use DBMS. In fact, the business may use more than one database of the same or different types when performing some tasks. In a certain case, an end user needs a specific data from all the databases that requires time and effort for obtaining the data. So far, there are many database tools available for data merging and conversion. However, most of them require the complete table to be merged or converted [2-7]. Data access in a heterogeneous environment can be performed through application program interface (API) [8-9] such as Open Database Connectivity (ODBC) [10] or Java Database Connectivity (JDBC) [11].

Recently, users usually depend on database management when dealing with many database applications in the same organization, especially when those database applications are implemented at different time to perform specific tasks. A specific driver is needed to configure a connection for each database application. The end user needs more time and effort to retrieve a particular data from more than one database applications and sometimes it is hard to retrieve the precise data. Lacking of the flexibilities in handling such transactions lead to this research.

This paper proposes a design of a prototype

architecture for providing flexibilities in database conversion. In addition to handling a whole table of a database, it is also possible to select part of the database to be converted into another database type. The prototype will consist of a master database that will import any table attributes. Furthermore, data can be imported from multiples databases of similar or different types. In this way, a user can perform specific task that will save time of analyzing as well as designing a new database system. On top of that, it will allow specific data from different types of the existing database to be imported and restructured automatically. With this centralized management, database access and integrity can be performed properly and automatically. Thus, more time and effort will be saved for analyzing, designing new database and performing data entry.

This paper is organized as follows: Section 2 discusses on the related work of issues on databases, their revolution and data manipulation. Section 3 presents the comparative study on database conversion tools and techniques. Section 4 illustrates the propose architecture, before the concluding remarks in Section 5.

2 Database

Databases are used to store, retrieve and manipulate data in many organizations. Database technology is used by individual (on personal computer) or group (on network for centralized or distributed system). The word "data" can be defined as facts that could be recorded and stored on a computer [12]. It can be processed to become information which is more convenient and suitable for communication, interpretation or processing by the end users. Moreover, data can also represent facts concerning people, places, events, objects or concepts. When data is grouped together and organized in a formalize manner, it can be called a database [1]. A database management involves both defining structures for storing information and providing mechanisms for manipulating the information [13]. There are many different types of DBMS, ranging from small systems that run on personal computers to large systems that run on mainframes.

DBMS can also be divided into three types which include relational, network and hierarchical. The most widely common type of DBMS today is the Relational Database Management Systems (RDBMS) [14]. Some DBMS may be better suited than others for specific types of applications. DBMS differ according to the types of logical data structures they support for representing information as data and the operations that rely on the DBMS languages [14].

RDBMS are powerful and often easy to operate. They have been used successfully in many types of organization, e.g. in business, manufacturing, and service industries, education, government and scientific research [15]. However, there are also many applications for which RDBMS provide a poor solution. In particular, RDBMS are poor at representing information about entities with complex structure. They are also deficient in handling modifications of data that involve complex operations [15].

A new generation of database technology is now emerging which can overcome many of the limitations of relational database technology. This new generation primarily supports two types of database, called Object Databases and Object-Relational Databases [14]. Object database management systems (ODBMS) and object Relational database management systems (O-RDBMs) are based on features of object-oriented programming languages. The combination of object oriented programming capabilities and conventional DBMS facilities provided by object and object-relational database technologies has produced a powerful environment for representing and making use of information about an organization [15].

2.1 Structures

The descriptions of data structures can also be called schemas. The database has two parts; database intension and database extension. Database intension is a set of schemas which define the structure of the database, whereby the database extension is the data values in the database [16]. Data within a database system is complex and needs to be modeled to gain better understandings [17]. Data model is easier to work with as compared to low level data such as bits, numbers and strings.

Data model is a method to describe data objects, relationship between data and consistency constraints by collection of conceptual tools. Its conceptual description of problem space is used in database design to express the model in term of entities, attributes, domains, and relationships. Thus, there is a need to have a basic architecture to build a model [17]. It helps users to view the data in term of its physical item, its usage and its relation among items. This will affect the output without the need to reuse the output value.

A databases management system (DBMS) architecture concerns on describing the location of all pieces of information that make up the database application [13]. In fact, it is suggested by the American National Standards Institute/Standards Planning and Requirements Committee (ANSI/SPARC) [18] to have three-level architecture model for data viewing. Figure 1 shows interaction of those levels; external, conceptual, and internal.

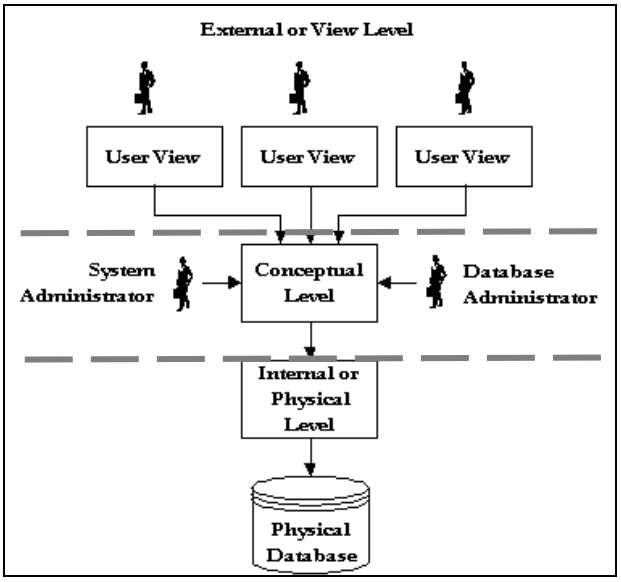

Figure 1. Three level ANSI/SPARC architecture [18]

Details of the models are as follows:

- *External* - individual user view: Number of external schema or user view in terms of real world such as students, lectures, invoices, and etc.
- *Conceptual* - community user view: this level is known as bridging the gap between internal and external levels. Thus, the conceptual scheme hides the details of the physical storage structure and concentrates on describing the entities, relation and constraints.
- *Internal* - physical or storage view: this level concentrates on the way of how data is stored on physical database or focuses on the database structure.

The *internal* model describes how data is physically represented while the *conceptual* model describes the logical structure of the database. The *external* model describes logical data structure that has been defined for specific applications [16].

The feature of the logical view in the database is made independent based on how the data is stored and it is called data independence. At the same time, it acts as a source to provide benefits for database technology [14]. Several benefits of the data independence are as follows [16]:

- The database may be altered without affecting users. For instance, in order to make a database system run faster, its existing algorithms or linking methods may be changed or new indexes may be introduced. Thus, the database may evolve without interfering the existing applications.
- Data may be shared by existing or newly created applications.
- Greater productivity is possible. Application programs are simpler due to exclude the details of how data is physically stored. Typically, a database system includes tools to automatically generate application programs from descriptions of what they must do, rather than how they must do it.
- Greater security and integrity control are possible. When database approach is applied, data is treated as a central resource that must be managed. This management is performed by a database administrator (DBA).

2.2 Connectivity

Database Connectivity is the method of interaction between database schema and database application as shown in Figure 2. Application developers from all areas need a specialized application domain system interface together with database management system called Application Programming Interface (API). API is a set of routines, rules, protocols and tools for building software applications and normally written by the enterprise company application developer. API eases the process of developing a database application by providing library calls from the third party vendor. Examples of standardized database connectivity interface are Java Database Connectivity (JDBC), Open Database Connectivity (ODBC) and Object Linking and Embedding Database (OLE DB).

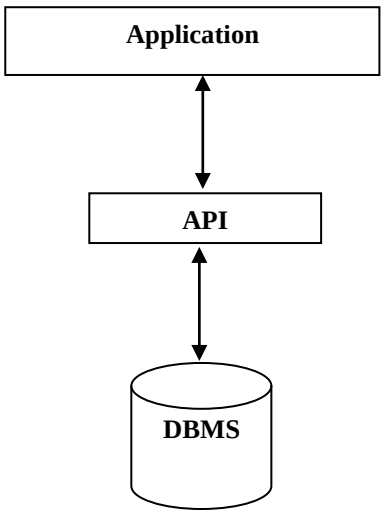

Figure 2. Database communication through Database connectivity (API)

Nowadays, most of the database systems are part of Client/Server architecture that is linked through API for connectivity. ODBC is one of the popular API that allows the developer to write database application to access a database schema using Structured Query

Language (SQL) [19]. It was released as a standard in 1992 resulting from the partnership between *Microsoft* and *Simba Technologies*. It provides components such as library of ODBC function calls, SQL syntax, a standard set of error codes, a standard way to connect and DBMS logging systems, a standard data types representation and a standard way for data conversion. There are four main components in ODBC architecture as follows [20] :

- application
- ODBC driver manager
- ODBC driver
- data source

The ODBC driver manager loads and unloads any application that is requested by the drivers and process some of ODBC function calls. The ODBC driver processes most of the ODBC function calls before submitting SQL requests to a specific data source. Next, it returns the results to the application. The driver will also modify the requested application to adjust its data format to suit the data source. Figure 3 shows the ODBC architecture.

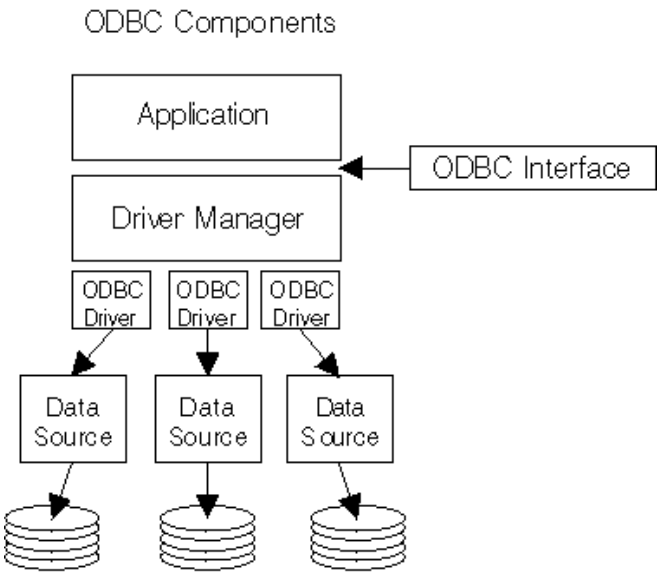

Figure 3. ODBC architecture [20]

2.3 Revolution

In the early invention of a computer, there was no database. During the first half of the 20th century, companies kept large numeric data on punched cards, and the data was retrieved and sorted by mechanical tabulating equipment. Database was introduced in the late 1960s and the file management system is the predecessor of database [12]. At this time, most computer stored data sequentially on magnetic tape. Most of the stored data are developed for mainframe computers, which could handle only a single data file and they were oriented toward specific data processing functions called File Management System (FMS).

The file management system (FMS) was also known as flat file database because the data stored in a single large file with interface between data files and programs. Data access is performed through sequential access methods. There were several disadvantages that include long time data searching process with the concept of indexing, data duplication and high maintenance costs. All these drawbacks led to the introduction of the hierarchical database system that was developed largely to manage data structure due to the difficulty in implementing the flat files and the increase demand in using the database systems [1]. Hierarchical database system was based on binary trees where data represents in logical form of upside down tree. Hierarchical database system enforces on one to many relationship between parent data and child data. The IBM created the first DBMS called information Management System (IMS) in the 1970s [13]. Its disadvantage lies on the restoration of the whole system when there is any modification.

From the hierarchical DBMS, many-to-many relationship was introduced and expanded to become a network data model. It represents more complex data relationship that could be modeled with hierarchical structure [13]. Network data model has sets of relationships for multiple parents. The relationship may represent one to many relations between an owner and a member. Each set has an owner record and a member record while the member record may also have several owners. This complex relationship creates system complexity and difficulties in designing and maintaining data that leads to lack of structural independence.

Later, the network and hierarchical system were improved by E.F. Codd (Edgar Frank Codd) in defining the relational model for databases and non-procedure data query [17]. The relational database allows data structures, storage and retrieval operations, and integrity constraints that organized data in tables and records with many fields or attributes. The system have became the database standard.

In 1985, Object-Oriented DBMS (OODBMS) was developed by introducing new methods of data organization with the goal to define object that could be reused in different programs for time saving and error reduction. It provides persistent storage for objects as well as query language, indexing, transaction support with rollback and commit [21-22].

2.4 Data Manipulation

Data manipulation across multiple heterogeneous databases can be performed through application program

interface (API) with standardized connectivity (e.g. ODBC, OLE DB, and JDBC). JDBC is an independent platform that provides Java databases applications and a database to manipulate data via method calls for built-in and query statement [23]. Figure 4 illustrates the JDBC architecture.

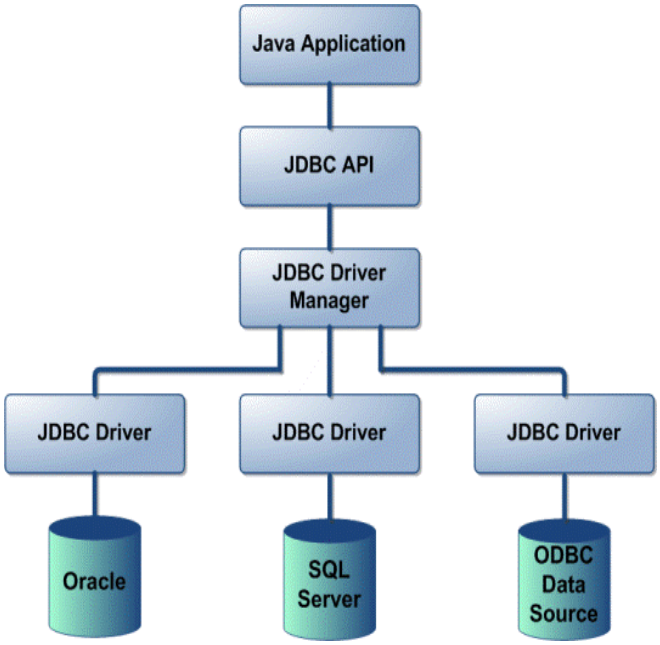

Figure 4. JDBC Architecture [11]

Much research has been undertaken to deal with multiple databases that include data integrity and conversion. In dealing with database integrity, validation and referential integrity are two most important elements for correct and reliable results. The validation usually involves complex and huge modern database applications that require powerful services for controlling the semantic correctness of data [24].

The database system itself is responsible to protect the database against incorrect data that is not expressed in the real world data or business rules [25]. In dealing with more databases, the link between two tables must be verified for the referential integrity. It is to ensure that references between data are really valid and intact [26]. As stated by Blaha [26], the referential integrity helps to improve data quality, faster development, fewer bugs and data consistency across applications.

In dealing with the database technique, many research works concern on four common issues that include migration, conversion, integration and merging. Database migration usually involves upgrading the database or system software. It is possible to be achieved through method migration and data migration [27]. Both method migration and data migration can be performed in distributive manner. Figure 5 illustrates the process structure in distributed environment.

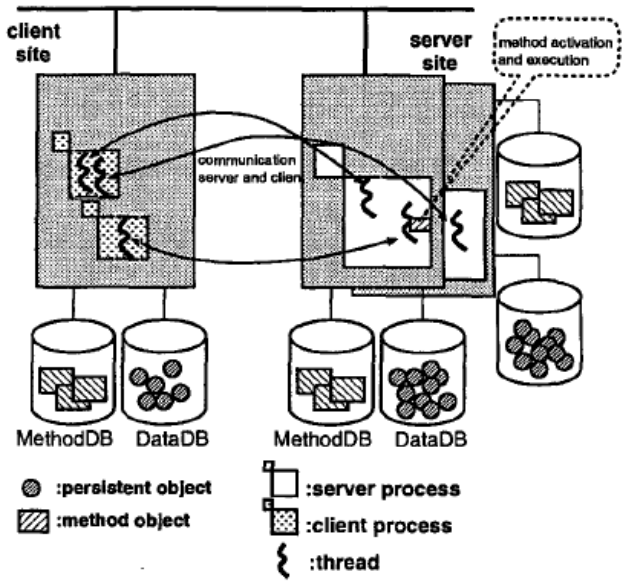

Figure 5. Process structure in distributed environment [27]

According to Meier [28], there are three migration strategies in commercial products. The strategies are data and code conversion, language transformation and data propagation between two database management systems. In data migration, the source data will be captured, followed by creating a relational database model, and the source data migration [29]. The flow of data migration is shown in Figure 6.

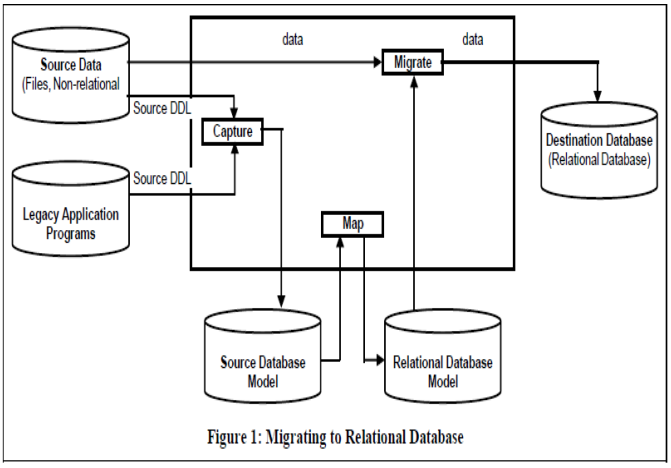

Figure 6. Migrating data to relational database [29]

There are four steps in the data migration process. All the steps must be followed accordingly as shown in Figure 7.

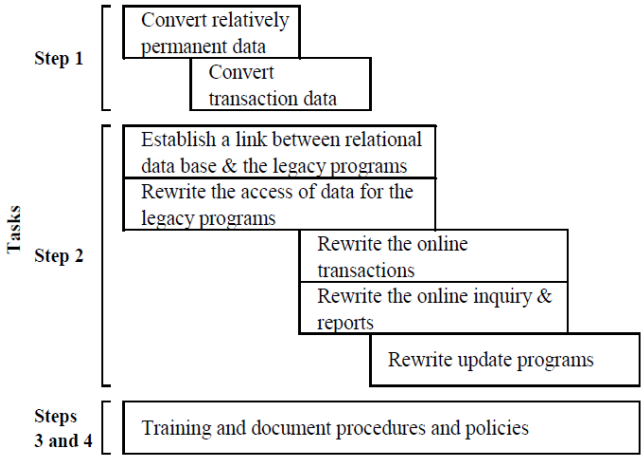

Figure 7. Steps in relational data migration process [29].

For database conversion, the focus is on converting data item values from one database to another database to improve productivity and flexibility [30]. Usually, the conversion also involves vendor to vendor for improving systems performance as well as getting more storage and applying new features of the provided products. Thus, there is a constraint on schema-equivalence for the first database (source) and second database (destination).

In database integration, a vital issue is data interoperability for the development of new information systems. The whole processing involves the existing data rather than data entering from different databases. Figure 8 depicts the global integration process [31].

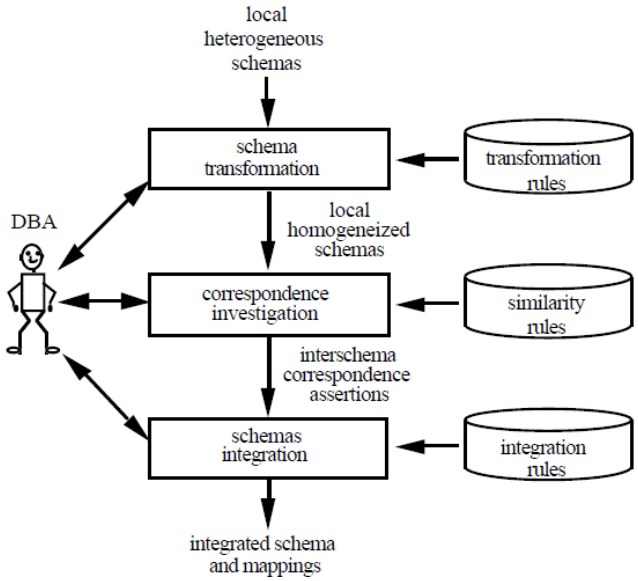

Figure 8. The global integration process [31]

There are two approaches to database integration that consist of data warehouse approach and federated database approach. The data warehouse approach emphasizes on data translation while the federated approach emphasizes on query translation [32]. The warehouse approach transforms data from diverse sources (distributed databases) into a local warehouse that will execute all queries and operations on the local data warehouse rather than the distributed places of source.

This approach overcomes several problems including slow response times and network bottlenecks as well as improving the query efficiency because it can be performed locally [32]. As for federated database approach, it uses middleware that works at the runtime to translate user query on a single federated schema into queries in local schemas [31]. To conclude, database integration is a collection of databases that are treated as one entity and viewed through a single user interface.

For database merging, large databases from various analysis functions usually gathered together to see their information relation between different databases. Therefore, managers and decision makers can select data from any databases and inspect multiple result lists [33]. In the merging process, both overlapping (e.g. usually names of individuals) and non-overlapping information are compared and combined [34]. Nevertheless, merging criteria of identical entity must be defined as accurate as possible. Figure 9 summarizes the integration process.

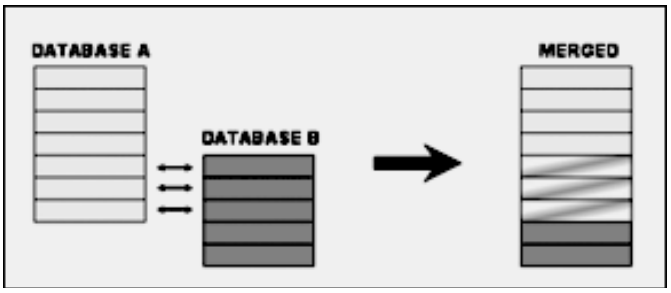

Figure 9. Integration of information contained in two databases [34].

In conclusion, the aim of all the related researches is to gather as much information as possible from multiple databases on the given targeted data.

3 Comparative Study

This section discusses comparative studies on database conversion tools and techniques in database management systems.

3.1 Database Conversion tools

Studies on eight selected database tools are conducted to analyze how database has been converted, migrated or merged by the tools. All of these tools are available

online and they can be downloaded and tested. Table 1 depicts the comparison between the tools [35].

Table 1. A Comparative Study on Database Conversion Tools

Tool	Creator	Purpose	Method
Full convert ent.[2]	Spectral Core	Database conversion	Database » database Schema » Schema
Data Sync [3]	Spectral Core	Find differences	Compare & synchronize
SwisSQL [4]	SwisSQL	Complete data migration	Migration schema & data
DatabaseSpy [36]	Altova	Migration structure	Compare & merge table
Database Merge [38]	Donald Fish Data	Database Merge	Combine records
Database Converter [5]	Sanmaxi	Data migration	Convert full database
Pro Data Doctor [6]	Pro Data Doctor	Database conversion	Convert full database
Dbconvert Line [37]	DMSoft	Data migration	Data import & export

Full convert enterprise tool, developed by *Spectral Core* is to do full conversion of sophisticated database. The old version of database is transformed into a new version [2]. The software has greatly reduced the time and effort required to convert the data. Figure 10 shows example of *Full convert enterprise tool*.

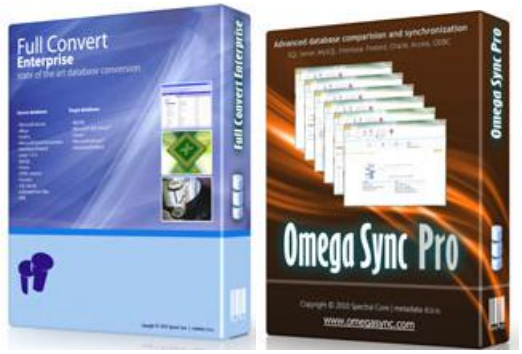

Figure 10. Full convert enterprise tool [2].

Another tool by the same company, *Data Sync*, has different approach by finding the differences in database tables and later, helps to synchronize these tables [3].

Other company, *SwisQL* [4], developed a complete data migration tool that helps the migration and transfer of database schemas and data across leading databases such as Oracle, IBM DB2, MS-SQL Server, Sybase, SAP DB, MySQL, PostgreSQL and MS Access. During the migration process, the user can change the table structure and the content of table. However, the migration is limited to a single database as a source [37].

Altova, a creator of XMLSpy and other leading XML, database, and UML tools, has developed a tool called *DatabaseSpy 2010* for easily compare and merge contents of database tables [36]. It can do migration on the structure of existing database tables from one database type to another. It also supports multiple database types and automatically adapts its database structure conversion functionality to many database types including Microsoft® SQL Server® 2008, IBM DB2 for iSeries® v5.4, 6.1, Oracle® 11g and MySQL®5. Figure 11 depicts the screenshots of *DatabaseSpy*

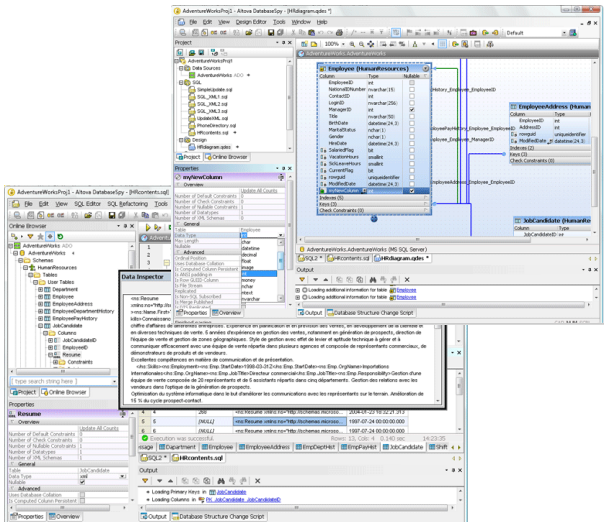

Figure 11. Screenshots of *DatabaseSpy* by Altova [36]

Another company, Donald Fish Data Management, provides database merges service that combines records from two or more lists into one relational database or flat file. When the same record occurs in more than one list, similar to duplication elimination, the multiple records are merged into one record [38]. The *Database Converter* [2] and *Pro Data Doctor* [6] convert full database. The former tool concerns on records within MSSQL-MySQL, MySQL-MSSQL, MSSQL-MS Acces and MySQL-MSSQL, while the latter tool runs on windows operating system for MSSQL-MySQL database converter, MySQL-MSSQL database converter and MS Access-MySQL. Other tool, *Dbconvert Line*, performs database migration tools with specialization on cross-database migration between multiple databases [39]. The example of its screenshots is shown in Figure 12.

Figure 12: Database converter sample screen shots [39]

3.2 Techniques in Database Manipulation

As data in the database has to be updated overtime, it requires more storage and processing power. The fast growing technology offers these requirements to be fulfilled and appropriate techniques must be used. Table 2 depicts the most popular techniques such as migration, conversion, integration and merging with the methods used by the researchers [35].

Table 2. Comparison between database techniques

Technique	Method	Research work
Migration	Data Types. Scheme . Language transformation. Data propagation.	Meier[28] Yoshida[27] Lin[29]
Conversion	Data attributes. Relational to object (schema). Data types schema.	Fong[30]
Integration	The global integration process. (Pre-integration, Correspondence identification Integration) Bio Meta Search Engine General Architecture .	Parent [31] Kazemian[39] Cheung[32]
Merging	Attributes Data Without change structure of database overlapping and non-overlapping information	Calvé [33] Wagner[34]

It can be summarized that in database conversion between different database types, the conversion process involves the whole database schema from one database type to another. For database migration, most of the processes are about migrating the current database to a new version. Sometimes, the migration process involves data conversion between data types and database constraints. In all cases, they require the same structure of database to do the migration. In database integration, combining different databases to one unit database is limited to the same database schema. In other words, the copy and transfer of data is onto one physical unit.

4 System Architecture

This study proposes a design of new prototype architecture on a database conversion with flexibilities and automatic restructuring. It will allow users to select some or all attributes and entities from different databases from a single database. This prototype will construct a master database automatically dependent on the user's request from the existing databases. Figure 13 illustrates the system architecture [35].

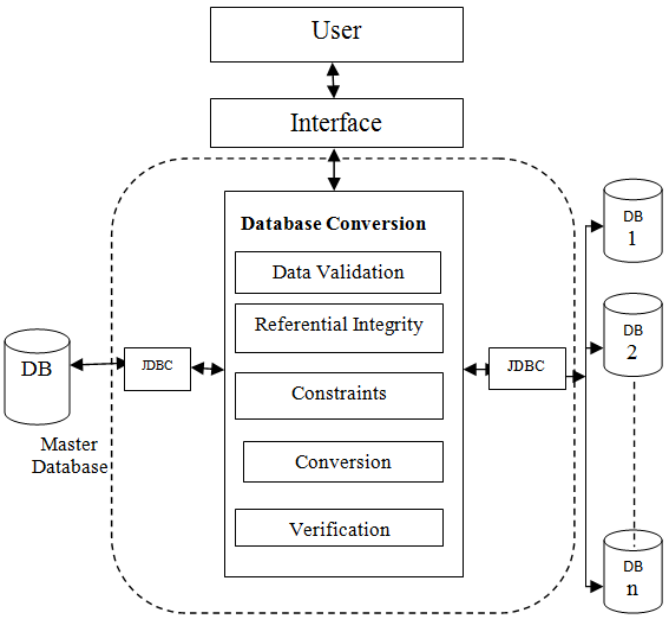

Figure 13. System architecture for flexible conversion

The system architecture is the conceptual design of six important components. The components are as follows:

1) *User*: User in this project is responsible about browsing the databases and selects the attributes or entities from different databases which require designing new database based on the previous databases.
2) *JDBC*: The way to access the database and manipulate data from any platform.
3) *Database Conversion*: The process of converting data. It is the most important practical components of this architecture, which is carried out automatically by the system. Therefore, the process for confirming, maintaining relationships and constraints as well as checking the validation of data between entities in the database are automatic. The validation is based on what the user has made in the selection of columns and tables for designing a new database. There are five elements to perform the database conversation process successfully. The elements are data validation, referential integrity, constraints, database conversion, and verification.
4) *Graphical User Interface*: Graphical user interface (GUI) is the platform for interaction with the user in order to display the databases. It is a way to access the databases as well as to do automatic design of a new database based on the user's selection.
5) *Single or Multiple Databases* : Single or multiple databases which are already existed in the organization in the form of similar or different types (known as the source databases). The end-user can

select tables and columns from these databases. Therefore, the data transfer is performed after a new database is designed by the program.

6) *Master Database*: Master database is the new database (target database) which is design automatically by the program based on the previous databases.

The flow of the proposed system will be undergone a few steps which is shown in Figure 14. Users are allowed to browse multiple databases through the user friendly graphical user interface (GUI).

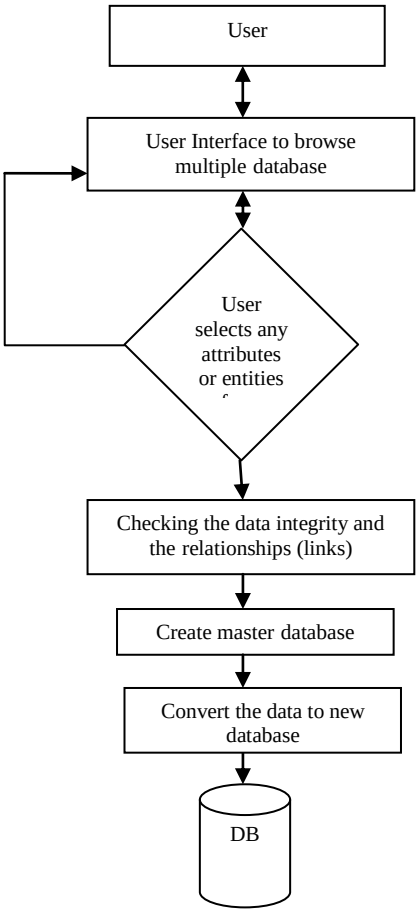

Figure 14. The workflow

Based on the displayed databases, users can make selection on the attributes and entities from the desired databases. Then, the system will perform automatic checking for data and its links between the attributes, constraints and data types before creating the master database. When the master database is created, the data conversion is ready to be performed.

5 Conclusion

In this paper, we have presented a comparative study on database conversion tools and techniques in database management. All the database conversion tools provide data sharing, migration and merging within the existing table with similar attributes. Therefore, we propose a new system architecture for database conversion which provides flexibilities in the record length and number of attributes within a table. In addition, JDBC connectivity will be used for an open platform. With such approach, data sharing, conversion, migration and integration could possibly be achieved. The added feature of automatic restructuring would ease users to upgrade and expand their existing database system without the need to incur cost in developing a new database system to cope with the current demand of database management and advances in computer technology.

References:

[1] Fred R. McFadden, Jeffrey A. Hoffer, *Modern database management*, Oracle Edition, Prentice Hall, 1999

[2] Full Convert Enterprise, spectralcore , http://www.spectralcore.com/fullconver/,2010

[3] Data Sync, spectralcore, http://www.spectral core. com/datasync,2010.

[4] swissql-data migration tool 5.7, SwisSQL, http://www.swissql.com/products/datamigration/data-migration.html,2010.

[5] Database converter, Sanmaxi, http:// www.database-converter.com,2009

[6] Database Conversion Tool, Pro Data Doctor, http://www.prodata doctor.com,2009.

[7] Oumtanaga S., Tiemoman K., Kimou P. and Florent D. N., Proposition of a model of Data backup in XML format, *6th WSEAS Int. Conf. on Software Engineering, Parallel and Distributed System*, 2007.

[8] Maydene Fisher, Jon Ellis and Jonathan Bruce , *JDBC API tutorial and reference*, Sun Microsystems, 2003 .

[9] Ali A. and Belal M., Replicas Redistribution in Distributed Database using Creative Evolutionary Systems Approach, *5th WSEAS Int. Conf. on Applied Computer Science*, 2006.

[10] Richard D.Hackathorn, *Enterprise Database Connectivity*, Wiley professional computing, 1993.

[11] JDBC Architecture, developersbook, http:// www.developersbook.com/jdbc/interview-questions/jdbc-interview-questions-faqs.php,2007

[12] Fred R.Mcafadden, Jeffrey A.Hoffer, *Database Management*, Benjamin/Cumming Publish Company, 1985.

[13] S.Sumathi. S. Esakkirajan, Fundamental of Relational Database Management System, *Springer*, 2007.

[14] Barry Eaglestone and Mick Ridley, Web Database Systems, *Mc Graw Hill*, 2001.

[15] Akmal Chaudhri, Mario Jeckle and Erhard Rahm,

Web, Web-Services, And Database Systems, *Springer*, 2002.
[16] Jing T. Yao, Web-Based Support Systems, Springer, 2010.
[17] RL Worrender, Database, crucial, 2003.
[18] Dbmaker,http://www.dbmaker.com.tw/reference/manuals/tutorial/tutorial_03.html,2002.
[19] Robert Signore, John Creamer, Michael O. Stegman, the ODBC Solution: Open Database Connectivity in Distributed Environments, Mcgraw-Hill, 1995.
[20] Richard D.Hackathorn, Enterprise Database Connectivity, Wiley professional computing, 1993.
[21] Rao, B.R. *Object-Oriented Databases: Technology, Applications, and Products*, McGraw-Hill. New York, 1994.
[22] Astrova I. and Kalja A., Mapping of SQL Relational Schemata to OWL Ontologies, *6th WSEAS Int. Conf. on Applied Informatiocs and Communications*, 2007.
[23] Suzanne W. Dietrich, Susan D. Urban and Ion Kyriakides, JDBC Demonstration CoursewareUsing Servlets and Java Server Pages, *ACM New York*, NY, USA, 2002.
[24] Paul W.P.J. Grefen and Peter M.G. Apers, Integrity control in relational database system, *Elsevier Science Publishers* B. V. Amsterdam, the Netherlands, the Netherlands, 1993.
[25] Can Turker, Michael Gertz, Semantic Integrity Support in SQL-99 and Commercial (Object-) Relational Database Management Systems, *Springer-Verlag New York*, Inc. Secaucus, NJ, USA ,2001 .
[26] Michael Blaha, *Referential Integrity Is Important For Databases*, Modelsoft Consulting Corp, 2005.
[27] Yusuke Yoshida, Masayoshi Aritsugi and Yoshinari Kanamori, Performance Evaluation of Combining Data Migration and Method Migration in Object Database Environments, *Australian Computer Society*, Inc, Darlinghurst, Australia, Australia, 2002.
[28] Andreas Meier, *Providing Database Migration Tools*, Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, 1995.
[29] Chang-Yang Lin, *Migrating to Relational Systems: Problems, Methods, and Strategies*, Contemporary Management Research, 2008.
[30] Joseph Fong, Converting Relational to Object-Oriented Databases, *ACM* New York, NY, USA, 1997.
[31] Christine Parent and Stefano Spaccapietra, Issues and Approaches of Database Integration, *ACM* New York, NY, USA, 1998.
[32] Kei-Hoi Cheung, Andrew K. Smith , Kevin Y.L. Yip, Christopher J.O. Baker and Mark B. Gerstein, Semantic Web Approach To Database Integration *In The Life Sciences, Elsevier Science Publishers* B. V. Amsterdam, the Netherlands, 2006 .
[33] Anne Le Calvé and Jacques Savoy, *Database Merging Strategy Based On Logistic Regression*, Pergamon Press, Inc. Tarrytown, NY, USA, 2000.
[34] H. Daniel Wagner, Tombstone Identification through Database Merging, 2008.
[35] Abidin S. Z. Z., Ahmad S. and Yafooz W. M. S., Towards Flexible Database Conversion with Automatic Restructuring, *14th WSEAS Int. Conf. on Computers*, 2007.
[36] DatabasSpy, Altova, http://www. altova.com /databasespy.html,2010.
[37] DBConvert Product Line, DMSoft Technologies, http://dbconvert.com,2009.
[38] Database Merge , Donald Fish Data Management, http://www.donaldfish.com/ database-merge.html, 2010.
[39] M. Kazemian, B. Moshiri, H. Nikhbakh and C. Lucas, Architecture for Biological Database Integration, *Artificial Intelligence and Machine Learning 2005 Conference (AIML 05)* ,2005.