A Generic Menu-Based Interface for Web Querying

By

Suhair Mohammad Al-Haj Hassan

Faculty of Information Technology
Middle East University for Graduate Studies

Supervisors:

Dr. Musbah Aqel

Prof. Munib Qutaishat

**A thesis submitted in partial fulfillment
of the requirements for the degree of Master of Science
in Computer Information Systems**

Amman, Jordan
June, 2008

جامعة الشرق الأوسط للدراسات العليا

إقرار تفويض

أنا ، سهير محمد الحاج حسن ، أفوض جامعة الشرق الأوسط للدراسات العليا بتزويد نسخ من رسالتي للمكتبات أو المؤسسات أو الهيئات أو الأشخاص عند طلبهم حسب التعليمات النافذة في الجامعة.

التوقيع: Suhair

التاريخ: ٤/٦/٢٠٠٨

Middle East University for Graduate Studies
Authorization Statement

I, Suhair Mohammad Al-Haj Hassan, authorize Middle East University for Graduate Studies to supply copies of my thesis to libraries, establishments or individuals upon their request, according to the university regulations.

Signature: Suhair

Date: June 4th, 2008

Committee Decision

This Thesis (A Generic Menu-Based Interface for Web Querying) was successfully defended and approved on June 4th, 2008.

Examination Committee Signature

Dr. Musbah Aqel
Associate Professor Department of Computer Information Systems
(Middle East University for Graduate Studies)

Prof. Sattar Aboud
Professor Department of Computer Information Systems
(Middle East University for Graduate Studies)

Prof. Munib Qutaishat
Professor Department of Computer Information Systems
(Petra University)

Dr. Fawaz Al-Zghoul
Associate Professor Department of Computer Science
(Jordan University)

Declaration

I do hereby declare the present research work has been carried out by me, under the supervision of Dr. Musbah Aqel and Professor Munib Qutaishat. And this work has not been submitted elsewhere for any other degree, fellowship or any other similar title.

Date: June 4th, 2008

Suhair Mohammad AL-Haj Hassan

Suhair

Department of Computer Information Systems

Faculty of Information Technology

Middle East University for Graduate Studies

DEDICATION

To the greatest person in the whole word, to my father, who taught me how to navigate life's many roads, and to my beloved mother for their love, encouragement, patience and big support. Without them nothing of this would have been possible.

To my dear brothers Osama, Amjad and Ammar, and to my sweet sister Nisreen for their love and encouragement.

To my father, mother, brothers, and sisters in law for their kindness and encouragement.

To my best friend Ayah and her husband, for their kindness and support.

And specially, to the Love of my life and my soul mate, my precious Fiancé Ahmad, for his love, support, encouragement, and being always there for me.

You all make the journey worthwhile… Thank you

ACKNOWLEDGMENTS

I would like to express my sincere appreciation to my supervisors Dr. Musbah Aqel and Professor Munib Qutaishat for their guidance, support and motivation through out my research.

My appreciation is also given to all of the Information Technology faculty members at the Middle East University for Graduate Studies and I am particularly grateful to Professor Mohammad Al-Haj Hassan, for helping and encouraging my efforts during the thesis work.

Finally, I would like to take a moment to specially thank my Fiancé Ahmad Samhan, for his immense patience, emotional support and encouragement during my research.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

A Generic Menu-Based Interface for Web Querying

By

Suhair Al-Haj Hassan

Supervisors

Dr. Musbah Aqel

Prof. Munib Qutaishat

The World Wide Web is a fertile area that holds a very large amount of useful information. Millions of users access the Web daily searching for various types of needed information. Having the right information from the right interface is an important issue in the Web querying field. While searching the web, earlier users had to be aware of various index servers that were deployed on the web to achieve an efficient information retrieval. In order to solve this problem, Web query languages that rely on Structured Query Language (SQL) were introduced such as: Web Structured Query Language (WebSQL) and Web Object Query Language (WebOQL) to extract data from the web. Nowadays, users differ in their proficiency and background knowledge. For those users who are not familiar with formal query languages and others, in this research we have developed a Generic Menu-Based Interface (GMBI) that provides good support and guidance on query formulation driven by a grammar-based control structure, to assist users. GMBI generates legitimate query step by step as menu choices, provides the users with the SQL query related to the search process, and retrieves the result and the data needed from the database back to the users. This supports users in remembering the query syntax and metadata, which does not require data processing and database structure knowledge. Preliminary, testing result showed that this wok can simplify the job of the Web users and make their search more efficient in information context.

Keywords: WebSQL, Word Wide Web**,** Control Grammar, Data Extraction**,** Information Retrieval, Mapping tools, Generic Menu-Based Interface

ملخص

واجهة بينيّة عامّة مبنيّة على اللوائح لاستعلام الشبكة العنكبوتية

إعداد

سهير محمد الحاج حسن

إشراف

الدكتور مصباح عقل

الأستاذ الدكتور منيب قطيشات

تعد الشبكة العنكبوتية العالمية حقلا خصبا يحتوي على كمية هائلة من المعلومات المفيدة، حيث يلج ملايين المستخدمين إلى الويب يوميا بحثا عن أنواع مختلفة مما يحتاجون إليه من معلومات، وإن إيجاد المعلومة المناسبة من الواجهة المناسبة يعد أمرا مهما في حقل أنظمة إدارة قواعد البيانات، وأثناء البحث عن طريق الويب، كان يجب على المستخدمين الأوائل أن يكونوا على دراية بالخوادم المفهرسة المختلفة المجندة لخدمة الويب لاسترجاع المعلومات بشكل فعال، وبهدف حل هذه العقبة تم استحداث لغات لاستعلام الويب تعتمد على لغة الاستعلام المهيكلة (SQL) مثل : (لغة WebSQL ولغة WebOQL) لاستخلاص البيانات من الويب.

في هذه الأيام يتفاوت المستخدمون في الخبرة والخلفية المعرفية، لذلك إخترنا القيام بهذا البحث الذي يهدف إلى تصميم وبرمجة واجهة بينية عامة مبنية على اللوائح توفر دعم وتوجيه لتكوين جمل إستعلام مقيدة بنظام تحكم هيكلي مبني على قواعد محددة، حيث تقوم الواجهة البينية بتوليد جمل استعلام سليمة خطوة بخطوة كخيارات معروضة بلوائح، وتوفر جملة الاستعلام المتعلقة بعملية البحث للمستخدمين، وتسترجع أيضاً نتيجة الاستعلام والمعلومات المطلوبة من قاعدة البيانات. وهذا يدعم المستخدمين باستذكار تركيب جمل الاستعلام والمعلومات عنها، والتي لا تتطلب معرفة بكيفية معالجة البيانات وهيكلية قواعد البيانات. ونتيجة لذلك فإن عملنا سوف ييسر عمل مستخدمي الانترنت ويجعل استرجاعهم للمعلومات أكثر كفاءة وفاعلية.

CHAPTER 1

INTRODUCTION

The Internet is already more widely deployed than any other computing system in history and continues to grow rapidly. It contains lots of web sites dedicated to huge number of topics. With so many Web sites, it is easy to get lost while searching for specific information. The Internet and the World Wide Web (WWW) play an increasingly important role in people's private and professional activities. Searching the Web is an every day need to a large number and variety of people, and perhaps the most important thing to understand about searching for information on the Internet is the fact that it is more like being a detective than simply pressing buttons. When searchers approach the Internet, they look for information on a certain topic. Knowing the types of search techniques available and mastering some general search tips can make the search process more profitable.

In the same way that the detective at the scene of a crime gets an immediate impression of the events that transpired, and then sets out to find clues and evidences that support that initial impression, Web searchers use their assumptions about what will be found as clues in their search. At the same time that the crime detective keeps an eye open for any evidence that paints a different story, successful Web searchers are open to different types of information that they had not anticipated, resources and formats that were not expected.

1.1 AN OVERVIEW OF WEB QUERYING

The WWW is a large, heterogeneous, distributed collection of documents connected by hypertext links. The most common technology that had been used for searching the Web depends on sending information retrieval requests to "index servers" that index as many documents as they can find by navigating the network. One problem with this was that users must be aware of the various index servers (many dozens of them are currently deployed on the Web), of their strengths and weaknesses, and of the peculiarities of their query interfaces. A more serious problem is that these queries cannot exploit the structure and topology of the document network. As a solution for those problems a query language, WebSQL, was proposed, which takes advantage of multiple index servers without requiring users to know about them, and that integrates textual retrieval with structure [19].

WebSQL is an SQL-like query language for extracting information from the web. Its capabilities for performing navigation of web hypertexts makes it a useful tool for automating several web-related tasks that require the systematic processing of either all the links in a page, all the pages that can be reached from a given URL through paths that match a pattern, or a combination of both. WebSQL also provides transparent access to index servers that can be queried via the Common Gateway Interface (CGI) [65].

In order to make WebSQL available to all WWW users, a WebSQL user interfaces were designed. The input form can be used as a template for the most common WebSQL queries making it easier for the user to submit a query.

Lots of rich and valuable work was done in the field of Web querying. But in spite of this, several problems still face Web users in information retrieval. As those users differ in their proficiency and background knowledge, most of them are not familiar with formal query languages and face problems in having the right information from the right interface, which is an important issue in the Web querying field. Most database interfaces provide poor guidance to Web users on query formulation, burdening them to learn, and recall precisely the query language and database structure and have data processing skills.

In this thesis, we aim to simplify the job of the Internet users and make their work more efficient in information retrieval, by designing and implementing a Generic Menu-Based Interface (GMBI) that will provide the users with good support and guidance, and enable them to enter search queries using menus with a control grammar imposed on them, and retrieve their needed data easily.

1.2 PROBLEM DEFINITION

Web users differ in their proficiency and background knowledge and most of them are not familiar with formal query languages, as we mentioned earlier. Also, they face problems in having the right information from the right interface because most database interfaces provide poor guidance to Web users on query formulation. These problems that face Web users while retrieving information from the Web, have been our main motivation to instigate this research work.

The problem we focus on is designing a search form or interface that would provide good support and guidance on query formulation, for different users who seek information through the Web.

Our research aims to provide the needed information that users are looking for with suitable guidance, and to generate legitimate query step by step as menu choices that will help users through their search process.

Also, our research will focus on a specific domain. We have chosen Web pages that contain various publications, such as books, conferences, articles, citations, journals, magazines, newsletters and much more to be our domain of interest.

1.3 THESIS OBJECTIVES

The major objectives of our study are:

- Analyzing certain Web sites related to publications, extract Web data from them at different levels, and then convert the extracted data into a Relational Database (RDB).

- Developing a generic menu-based front-end interface that will provide support and guidance on query formulation driven by a grammar-based control structure.
- Generating legitimate query step by step as menu choices, and provide the user with the SQL statement that could work successfully with different types of databases.
- Displaying results to users which depend on the database we have generated from the different Web sites we have analyzed.

1.4 REQIREMENTS

In our work, we used certain tools and software applications in order to generate a database, and then develop our interface. In this section, we are going to present a brief description for those tools:

1.4.1 TOAD FOR ORACLE VERSION 9.0

Toad is a powerful, low-overhead database development and administration tool that makes database and application development faster and easier and simplifies day-to-day database administration tasks, by providing an intuitive graphical user interface to Oracle [53].

Toad is an industry-leading tool that increases user productivity and application code quality while providing an interactive community to support the user experience. Whether the user is a developer, application developer, Data Base Administrator (DBA) or business analyst, Toad offers specific features to make users more productive.

Leveraging Quest's ability to provide comprehensive cross-platform thought leadership, Toad enables users to extract greater value from heterogeneous database environment.

Toad also, makes PL/SQL development faster and easier. Advanced editors allow users to work on multiple files simultaneously even different file types such as SQL, PL/SQL, HTML, Java, and text. Hot keys, auto-correct, type-ahead, syntax highlighting, version control and numerous other productivity features speed development, while editing and testing are made easier with integrated result sets, explain plans, tracing, and DBMS_OUTPUT views.

Toad for Oracle is a comprehensive database tool that has evolved over the past 10 years into the most advanced and widely-used database tool of its kind. With simple navigation and seamless workflow for database development, as well as advanced PL/SQL editing, optimization and database administration capabilities, Toad proved to be the ideal tool for both the expert and the new Oracle user.

Using this tool, users can visually create queries and execute them, construct and manage database objects, share project assets such as templates, scripts, and code snippets, with other team members using Project Manager, Record workflow steps and play them

back later with Action Recall, and Profile their PL/SQL code to find performance bottlenecks and check for code coverage. Toad for Oracle also has its own team collaboration utility called Team Coding, which works with Toad's editor to control code access.

Toad for Oracle provides the following features and benefits:

- Greater efficiency when performing daily tasks.
- Greater accuracy when writing, debugging and tuning code.
- Flexibility to perform many development and administration tasks from a single tool.
- A smooth workflow that allows users to move from one task to another easily.
- Extensive reporting capabilities that produce quantifiable data and documentation.
- Access to Oracle expertise through various Toad user communities.

In our work, we used Toad for Oracle to create the tables we needed in oracle database. And then, we used the tool to make the import process, to import data from excel sheets to the tables created in the database, as will be shown and discussed in details in chapter 3.

1.4.2 ORACLE FORMS BUILDER 10g AND ORACLE DATABASE 10g

Oracle offers a complete and integrated set of application development and business intelligence tools that supports any development approach, any technology platform, and any operating system [51].

Forward-thinking architects and developers are using these tools today to address the complexity of their application and IT environments with Service-Oriented Architecture (SOA), which facilitates the development of enterprise applications as modular business services that can be easily integrated and reused, creating a truly flexible, adaptable IT infrastructure [51].

Oracle's development tools strategy is to offer software development tools that enable the development of enterprise applications on an Internet model. The software development tools consist of two parallel product offerings [25]:

- **Traditional development tools:** This solution set consists of the modeling environment provided by Oracle Designer, the application development framework provided by Oracle Forms, and the batch/ scheduled reporting solution offered by Oracle Reports. The programming language and execution environment for these tools is PL/SQL.
- **Java development tools:** Recognizing the rapid growth and adoption of Java/ J2EE technologies in the industry, Oracle also offers users the choice of a UML-based modeling environment and 3GL development within Oracle JDeveloper, an integrated J2EE Application Development Framework (Oracle ADF), and a Java-

based end-user reporting solution that will be provided in a future release of Oracle's business intelligence tools. The programming language for these tools is Java and the execution environment is any J2EE container, including Oracle's own Oracle container for J2EE (OC4J).

From the traditional development tools set, Oracle Forms, a component of the Oracle Developer Suite, is Oracle's long-established technology to design and build enterprise applications quickly and efficiently. Oracle remains committed to the development of this technology, and to the ongoing release as a component of the Oracle platform. This continuing commitment to Forms technology enables users to leverage their existing investment by easily upgrading and integrating existing Oracle Forms applications to take advantage of web technologies and Service-Oriented Architectures (SOA) [51].

Oracle Database 10*g* is the first database designed for grid computing, the most flexible and cost-effective way to manage enterprise information. It cuts costs of management while providing the highest possible quality of service.

In addition to providing numerous quality and performance enhancements, Oracle Database 10*g* significantly reduces the costs of managing the IT environment, with a simplified install, greatly reduced configuration and management requirements, and automatic performance diagnosis and SQL tuning. These and other automated management capabilities help improve DBA and developer productivity and efficiency.

In our work, we used Oracle Forms Builder 10g to design and implement our menu-based interface, and we tested it on Oracle Database 10g. **Figure 1.1** below shows the Object Navigator of Oracle Forms Builder 10g.

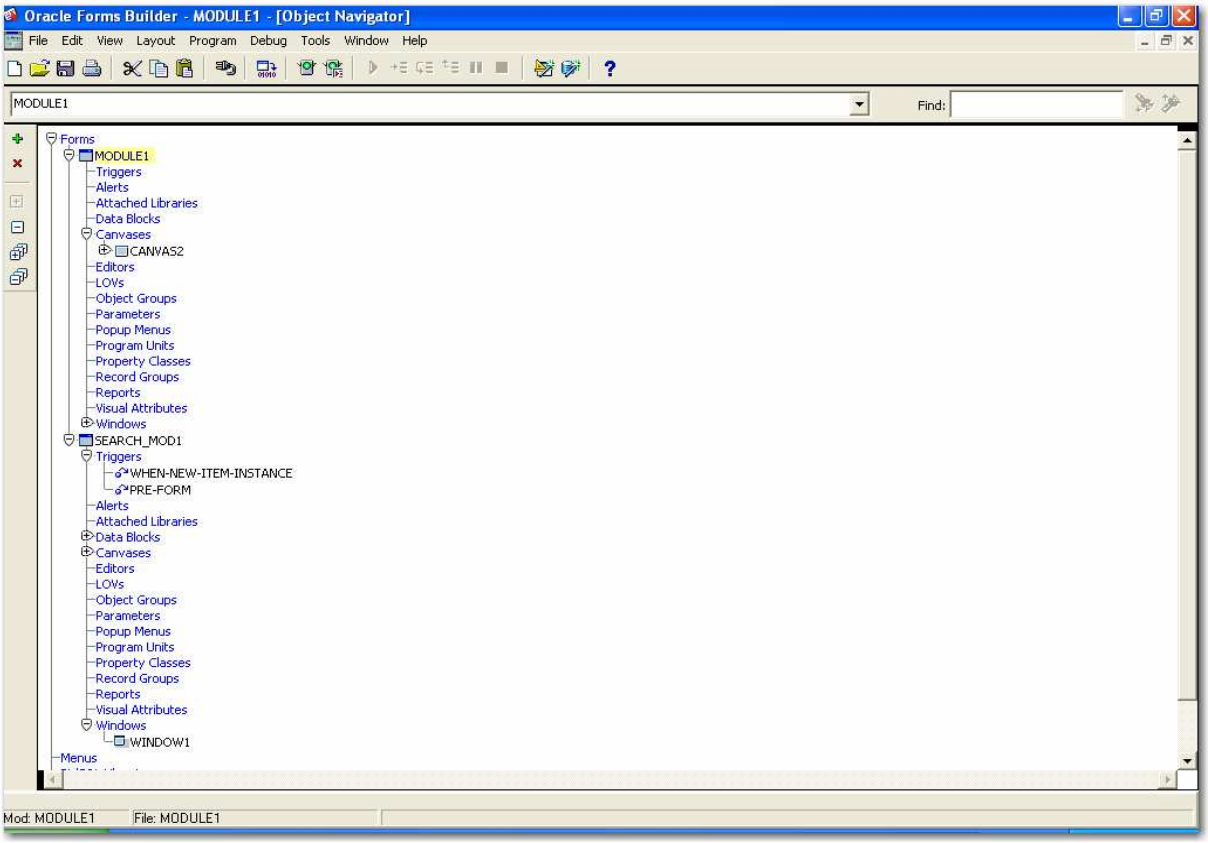
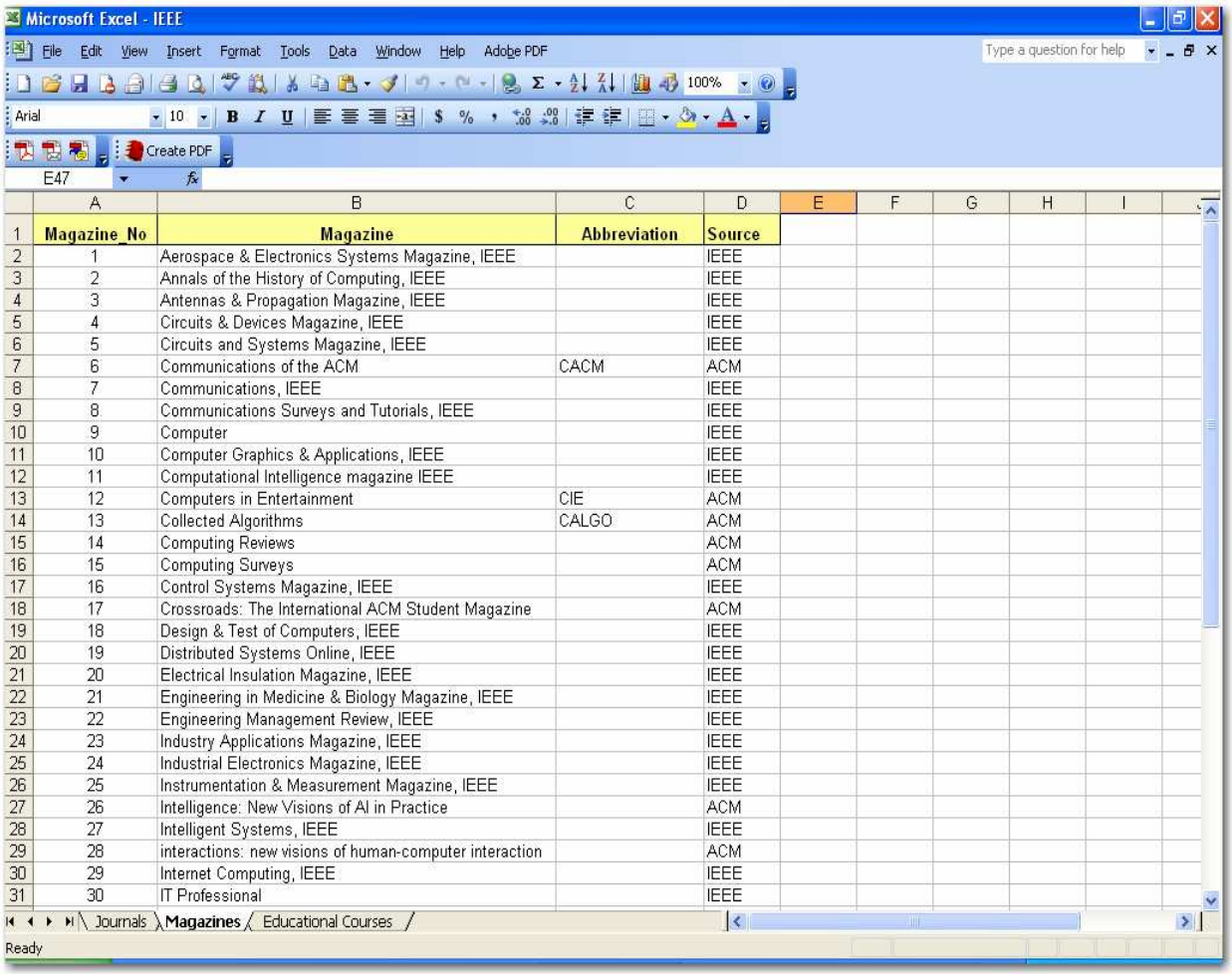


Figure 1.1: Oracle Forms Builder Object Navigator

1.4.3 MICROSOFT EXCEL

Microsoft Excel has been used to import the data needed from the websites and organize it into well-formatted tables to be imported to oracle database later on. A screen shot of one of the excel sheet is provided in **Figure 1.2** below.



Microsoft Excel - IEEE

Magazine_No	Magazine	Abbreviation	Source
1	Aerospace & Electronics Systems Magazine, IEEE		IEEE
2	Annals of the History of Computing, IEEE		IEEE
3	Antennas & Propagation Magazine, IEEE		IEEE
4	Circuits & Devices Magazine, IEEE		IEEE
5	Circuits and Systems Magazine, IEEE		IEEE
6	Communications of the ACM	CACM	ACM
7	Communications, IEEE		IEEE
8	Communications Surveys and Tutorials, IEEE		IEEE
9	Computer		IEEE
10	Computer Graphics & Applications, IEEE		IEEE
11	Computational Intelligence magazine IEEE		IEEE
12	Computers in Entertainment	CIE	ACM
13	Collected Algorithms	CALGO	ACM
14	Computing Reviews		ACM
15	Computing Surveys		ACM
16	Control Systems Magazine, IEEE		IEEE
17	Crossroads: The International ACM Student Magazine		ACM
18	Design & Test of Computers, IEEE		IEEE
19	Distributed Systems Online, IEEE		IEEE
20	Electrical Insulation Magazine, IEEE		IEEE
21	Engineering in Medicine & Biology Magazine, IEEE		IEEE
22	Engineering Management Review, IEEE		IEEE
23	Industry Applications Magazine, IEEE		IEEE
24	Industrial Electronics Magazine, IEEE		IEEE
25	Instrumentation & Measurement Magazine, IEEE		IEEE
26	Intelligence: New Visions of AI in Practice		ACM
27	Intelligent Systems, IEEE		IEEE
28	interactions: new visions of human-computer interaction		ACM
29	Internet Computing, IEEE		IEEE
30	IT Professional		IEEE

Figure 1.2: Excel Sheet Sample

During our work, we have tried other software packages, and get knowledge about them. We did not continue using them because we found others that serve our purposes in a better way. For the benefits of the readers of this thesis, we give a brief description of some of those packages in Chapter 2.

1.5 SIGNIFICANCE

This research work could mainly serve web searchers and internet users who seek particular information through the WWW, specially in the domain of several publications like: Books, Journals, Magazines, Conferences, Articles and much more.

Our research is also directed to DB users, developers and researchers who are interested in DBMS issues and also the Web querying field.

1.6 RELATED WORK

There has been lots of work and research in query languages, querying the WWW and menu-based interfaces. In this section we are going to introduce some of the most important related work in this field, which provides a good guidance to our work.

- Qutaishat [29] described an "Arabic Menu-Based Natural Language Interface to Database Systems" (AMID), which is a front-end interface that was conceived based on menus of lexical elements of the meta-data driven by a grammar-based control structure. Their methodology in constructing the system is a knowledge-based, meta-programmed expert system approach. AMID scope avoids the problem that users who are not used to formal query language (e.g. SQL) face in having the right information from the right interface because of poor guidance that most interfaces provide. AMID scope avoids this problem by guiding the user's query construction effectively and generating legitimate query constituents incrementally as menu choices. The Architecture of AMID is shown in **Figure 1.3**.

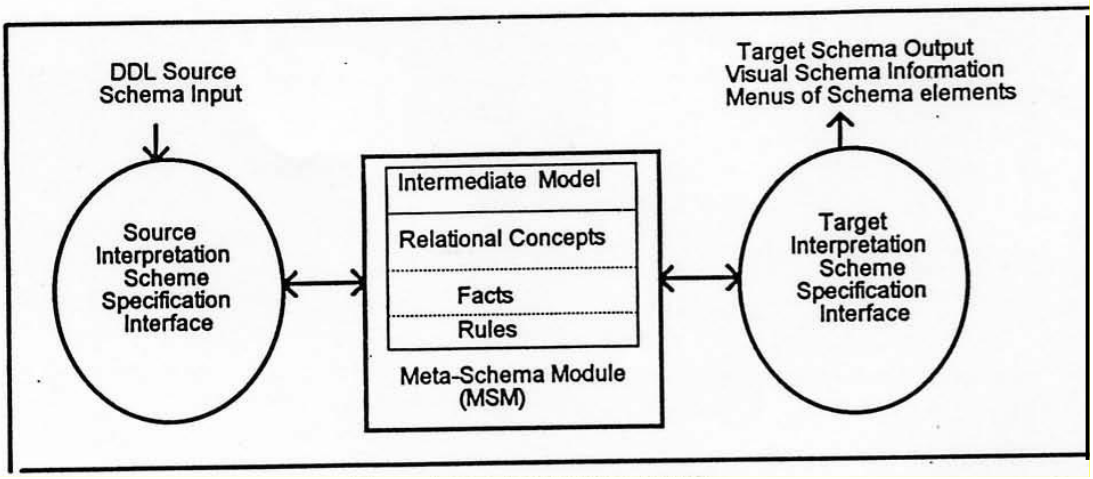


Figure 1.3: The Architecture of AMID [29]

- Konopnicki and Shmueli [14]. The W3QL approach is substantially different from the query language WebSQL. In W3QL approach, they emphasize extensibility and interfacing to external user-written programs and UNIX utilities. While extensibility is a highly desirable goal when the tool runs in a known environment, WebSQL tool can be downloaded to an arbitrary client and run with minimal interaction with the local environment. On the other hand, they support filling out forms encountered during navigation, and discuss a view facility based on W3QL, while WebSQL do not currently support either.

- Lakshmanan et al. [14]. Another recent effort in this direction is the WebLog language. Unlike WebSQL, WebLog emphasizes manipulating the internal structure of Web documents. Instead of regular expressions for specifying

paths, they rely on Datalog-like recursive rules. The paper does not describe an implementation or formal semantics.

- Mendelzon et al. [19] proposed a query language, WebSQL that takes advantage of multiple index servers without requiring users to know about them, and integrates textual retrieval with structure and topology-based queries. They gave a formal semantics for WebSQL using a calculus based on a novel "virtual graph" model of a document network. They proposed a new theory of query cost based on the idea of "query locality", that is, how much of the network must be visited to answer a particular query. They gave an algorithm for characterizing WebSQL queries with respect to query locality. Finally, they described a prototype implementation of WebSQL written in Java, as shown in **Figure 1.4**.

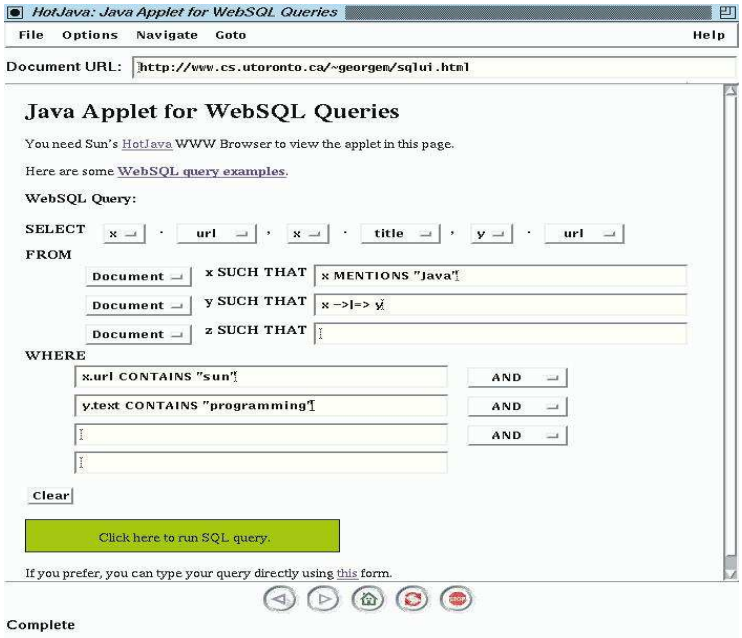


Figure 1.4: The WebSQL User-Interface [19]

- Arocena et al. [1]. In this paper they produced Applications of a web query language and reported on their experience using WebSQL, a high level declarative query language for extracting information from the Web. WebSQL takes advantage of multiple index servers without requiring users to know about them, and integrates full-text with topology-based queries. The WebSQL query engine is a library of Java classes, and WebSQL queries can be embedded into Java programs much in the same way as SQL queries are embedded in C programs. This allows accessing the Web from Java at a much higher level of abstraction than a bare HTTP request. They used WebSQL for applications related to Web maintenance and for the definition of the content

of domain-specific text indexes. Using the library, they have also implemented a client-server architecture that allows users to perform interactive intelligent searches on the Web from an applet running on a browser. They presented a WebSQL system architecture which demonstrates an implementation of the WebSQL compiler, query engine, and user interfaces, as shown in **Figure 1.5**.

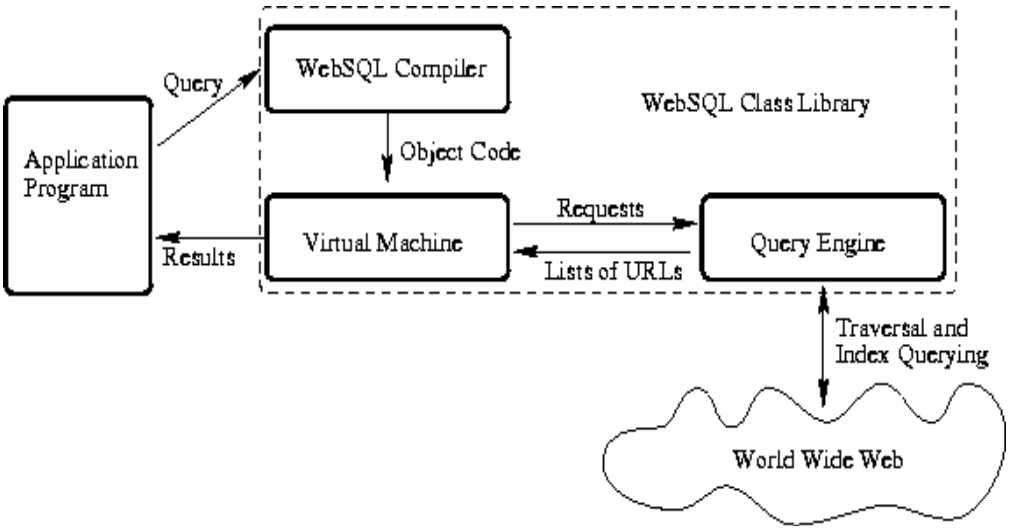


Figure 1.5: The Architecture of the WebSQL System [1]

They have also developed a much more user-controllable front-end under the form of a Java applet for Java-aware browsers, as shown in **Figure 1.6.**

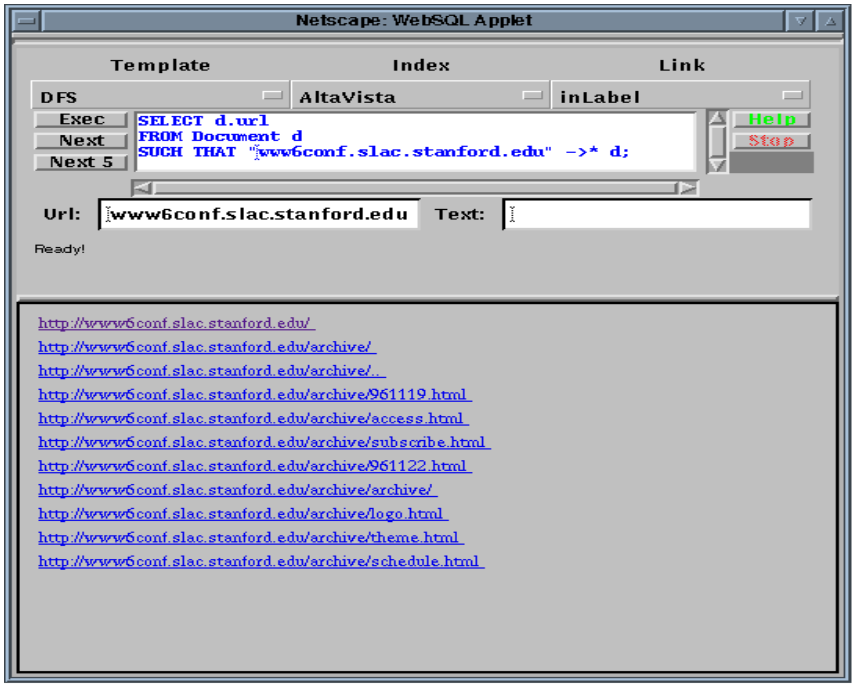


Figure 1.6: The WebSQL Applet [1]

- Patel et al. [26] presented an object-oriented database methodology for querying Web sources of data, structured in XML, is presented in this paper. A querying system is developed based on this methodology using ObjectStore database system. XML data are converted into objects that can be stored as part of the ObjectStore database system. Several querying interfaces for querying, searching and browsing the database are developed. The methodology presented in this paper is easily adaptable to any source of Web data that is represented in XML.

- Naughton et al. [24]. A truly useful system must provide mechanisms to find the XML files that are relevant to a given query, and deal with remote data sources that either provide unpredictable data access and transfer rates, or are infinite streams, or both. The Niagara Internet Query System was designed from the bottom-up to provide these mechanisms. It finds relevant XML documents by using a novel collaboration between the Niagara XML-QL query processor and the Niagara "text-in-context" XML search engine. To handle infinite streams and data sources with unpredictable rates, it supports a "get partial" operation on blocking operators in order to produce partial query results, and inserts synchronization packets at critical points in the operator tree to guarantee the consistency of (partial) results. **Figure 1.7** shows the high level architectural overview of the Niagara Internet Query System.

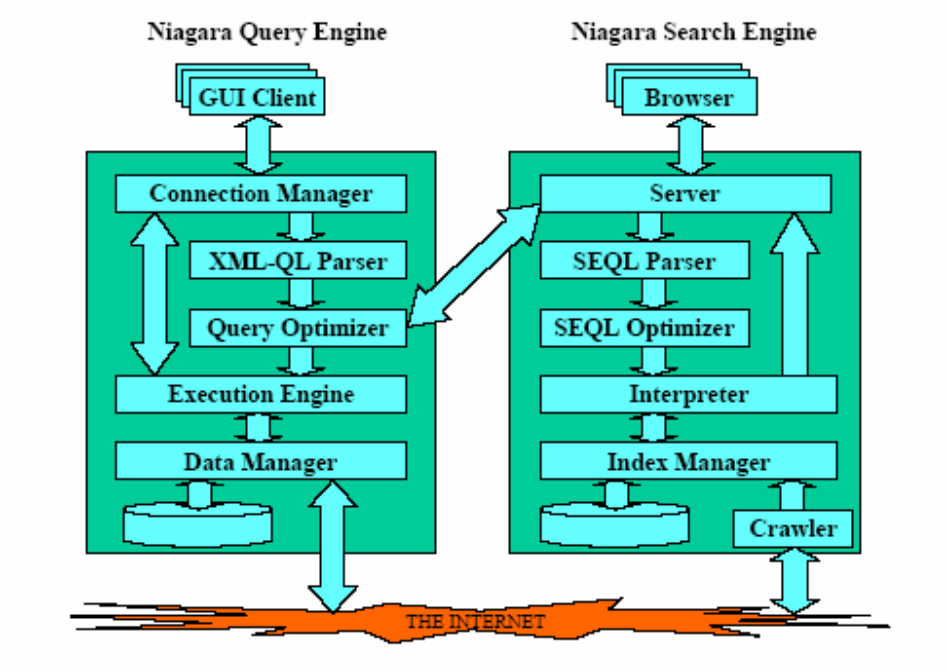


Figure 1.7: Architecture of the Niagara System [24]

- Bourret [5]. This paper discussed mapping Document Type Definitions (DTDs) to Databases. They discussed two mappings from XML to databases: a *table-based mapping* and an *object-relational (object-based) mapping*. Both mappings model the data in XML documents rather than the documents themselves. This makes the mappings a good choice for data-centric documents and a poor choice for document-centric documents. Both mappings are commonly used as the basis for software that transfers data between XML documents and databases, especially relational databases. An important characteristic in this respect is that they are bidirectional. That is, they can be used to transfer data both from XML documents to the database and from the database to XML documents.

- Srivastava et al. [35]. Web mining, i.e. the application of data mining techniques to extract knowledge from Web content, structure, and usage, is the collection of technologies to fulfill the potential of extracting valuable knowledge from the Web which has been quite evident. Interest in Web mining has grown rapidly in its short history, both in the research and practitioner communities. This paper provided a brief overview of the accomplishments in the field, both in terms of technologies and applications.

- Piotrowski [28]. A natural language interface for legal databases was presented in this paper. Browsing big legal databases has become a part of daily work activity in an almost every modern law office. Legal regulations change more often than it was in the past and to remain up-to-date, one must use electronic media collections for help. The problem is that most popular databases take queries in formal query languages (e.g. SQL), which can be

difficult for the casual database user. This project tries to determine, how the lawyers can be helped by providing natural language interface to legal knowledge databases. It can possibly separate a lawyer or any other user that looks for legal information, from complexity and understandability problems that formal query languages involve.

- Atay et al. [3]. Storing and querying XML documents using a RDBMS is a challenging problem since one needs to resolve the conflict between the hierarchical, ordered nature of the XML data model and the flat, unordered nature of the relational data model. This conflict can be resolved by the following XML-to-Relational mappings: schema mapping, data mapping and query mapping. In this paper, they proposed:
  - A lossless schema mapping algorithm to generate a database schema from a DTD, which makes several improvements over existing algorithms.
  - Two linear data mapping algorithms based on DOM and SAX, respectively, to map ordered XML data to relational data.

- Gracia et al. [11]. A Multiontology Disambiguation Method for Querying the Web was presented in this paper. The lack of explicit semantics in the current Web can lead to ambiguity problems. Though disambiguation is a very well-known problem in Natural Language Processing and other domains, traditional methods are not flexible enough to work in a Web-based context. In this paper they have identified some desirable properties that a Web-oriented disambiguation method should fulfill, and make a proposal according to them. The proposed method processes a set of related keywords in order to discover and extract their *implicit* semantics, obtaining their most suitable senses according to their context. The possible senses are extracted from the knowledge represented by a pool of ontologies available in the Web.

- Cafarella et al. [7]. The Web contains a huge amount of text that is currently beyond the reach of structured access tools. This unstructured data often contains a substantial amount of implicit structure, much of which can be captured using information extraction (IE) algorithms. By combining an IE system with an appropriate data model and query language, they enabled structured access to all of the Web's unstructured data. They proposed a general-purpose query system called the extraction database (ExDB), which supports SQL-like structured queries over Web text. They also described the technical challenges involved, motivated in part by their experiences with an early 90M-page prototype. The Schema Extraction Model they proposed, attempts to derive a single "best" schema for an input set of extractions. It then populates the schema with the extractions to generate a relational database that can be queried using standard SQL. The work flow for this model is shown in **Figure 1.8**.

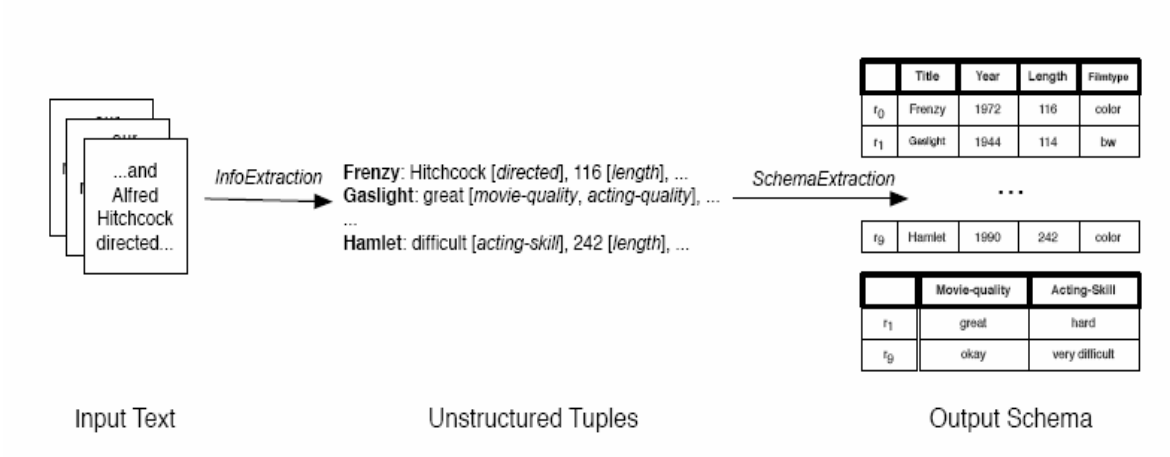


Figure 1.8: The Schema Extraction Model [7]

1.7 THESIS CONTRIBUTION

In this thesis, we analyzed different Web sites related to publications, and perform data and knowledge extraction from them in several levels. Then, our focus was to convert this data and import it into oracle database using Toad for oracle tool, which we are going to discuss later on.

For those users who are not familiar with formal query languages and others, this work aims to design and produce a Generic Menu-Based Interface (GMBI) that will provide support and guidance on query formulation driven by a grammar-based control structure. After the testing and evaluation phase of our front-end interface, we proved that the interface generates legitimate query step by step as menu choices, and provide the user with the SQL statement that could work successfully with different types of databases, such as: Microsoft Access, Microsoft SQL Server, Oracle, My SQL, Sybase, and IBM DB2. This supports users in remembering the query syntax and metadata, which does not require data processing and database structure knowledge.

Furthermore and most importantly, we proved that our interface also could provide users with the search result they are seeking successfully, and retrieve the needed data from the database. As a result, our work will simplify the job of the Internet users and make their work more efficient in information retrieval.

1.8 METHODOLOGY

Our methodologies for developing GMBI are based on experimental results and full implementation. To evaluate our interface, it is tested through three stages of testing process. Unit testing involves checking every program unit independently, to insure that it operates correctly. After that, the individual components are integrated and tested as a

complete system with simulated data. Finally, the whole system is tested with real data before it is accepted for operational use.

1.9 THESIS ORGANIZATION

In addition to several appendices, our thesis is divided into 5 chapters. In this section, we will describe briefly the content of these chapters.

Chapter 1: It is this introductory chapter, in which we have given an idea about the problem we have chosen for this work, the most relevant related work, the software used in this work, and the organization of the thesis.

Chapter 2: In this chapter, we have summarized various Web information mapping and conversion tools that convert and import Web data to different types of databases. Also, we have described different conversion phases and methodologies that could be used to make this process done.

Chapter 3: In this chapter, we have presented the process of converting Web information to relational database and describe it in details. First of all, we have presented the domain we have selected to be our focus, which are Web sites that concern publications. Also, we have discussed importing Web data to Microsoft Excel, including creating and customizing the Web query, and also modifying the Excel sheets to a well-organized format. Then, we have discussed the process of importing the data from Microsoft Excel to Oracle database, using 'Toad for Oracle' tool.

Chapter 4: This chapter is dedicated to present the development process of our Generic Menu-Based Interface (GMBI). We have mainly discussed its design, implementation and testing.

Chapter 5: Our final results have presented in this chapter. Also, we have presented our interface advantages and benefits. And finally, a comparison between our produced interface and other forms and interfaces has been discussed.

Chapter 6: Conclusions have been presented in this chapter. Also, we have presented our work contribution. And finally, some ideas for the future work have been given.

CHAPTER 2

WEB DATA MAPPING AND CONVERSION TOOLS AND METHODOLOGIES

While most critical enterprise data is stored in relational databases today, XML has become the technology of choice for data exchange and content management. As a result, the importance of working with XML data in harmony with relational databases is paramount [16].

In this chapter, we have introduced several methods and tools that convert back and forth between databases and XML files. Although eventually not used directly in our work, but mentioning them and their benefits will be valuable for the completion of our work and for the reader's benefit.

2.1 SITEMAPS

During our research, we found that one of the important issues related to searching the Web, analyzing its information and data specially in XML format and then mapping this data to relational databases is Sitemaps. In this section, we have presented Sitemaps, their importance, their benefits, and their different formats.

2.1.1 SITEMAPS AND THEIR BENEFITS

A Site map (or Sitemap) is a graphical representation of the architecture of a Web site [23]. It can be either a document in any form used as a planning tool for Web design, or a Web page that lists the pages on a Web site, typically organized in hierarchical fashion. This helps visitors and search engine bots find pages on the site [47].

Sitemaps are an easy way for Webmasters to inform search engines about pages on their sites that are available for crawling. In its simplest form, a Sitemap is an XML file that lists URLs for a site along with additional metadata about each URL (when it was last updated, how often it usually changes, and how important it is, relative to other URLs in the site) so that search engines can more intelligently crawl the site [58].

Web crawlers usually discover pages from links within the site and from other sites. Sitemaps supplement this data to allow crawlers that support Sitemaps to pick up all URLs in the Sitemap and learn about those URLs using the associated metadata. Using the Sitemap protocol does not guarantee that Web pages are included in search engines, but provides hints for Web crawlers to do a better job of crawling your site. It is easier to crawl a sitemap than a lot of pages [58].

Sitemaps can improve search engine optimization of a site by making sure that all the pages can be found. This is especially important if a site uses Macromedia Flash or JavaScript menus that do not include HTML links [47].

Most search engines will only follow a finite number of links from a page, so if a site is very large, the Sitemap may be required so that search engines and visitors can access all content on the site [47].

While some developers argue that Site index is a more appropriately used term to replay page function, Web visitors are used to seeing each of the two terms and generally associate both as one and the same. However, a site index is often used to mean an A-Z index that provides access to particular content, while a Site map provides a general top-down view of the overall site contents [47]. **Figure 2.1** presents the Sitemap of the ACM Web site.

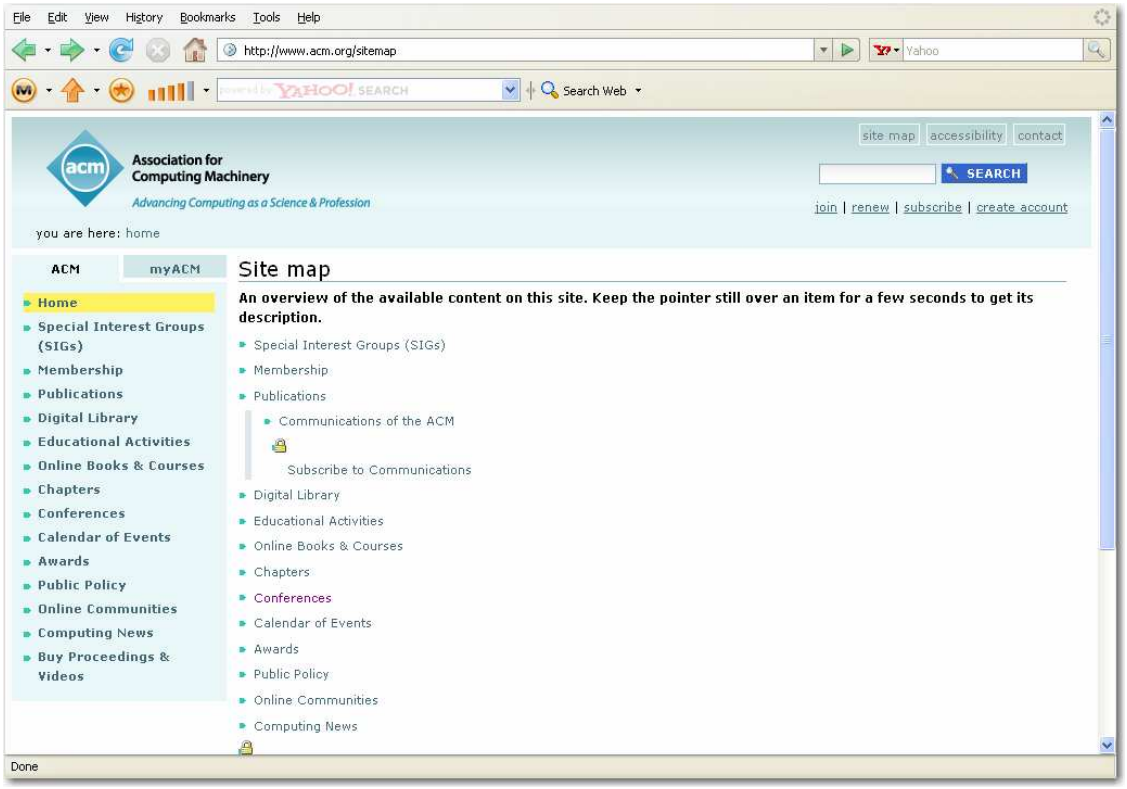


Figure 2.1: ACM Sitemap

2.1.2 SITEMAPS FORMATS

Sitemaps could be generated in various formats, which are: XML format, HTML format, Resources of a Resource (ROR), Really Simple Syndication (RSS) format, and Text format.

The XML format is now recognized by main search engines. It is intended to give various information to Googlebot and other crawlers. The XML Sitemap Protocol format consists of XML tags. All data values in a Sitemap must be entity-escaped. The file itself

must be UTF-8 encoded. The XML document is generated according to the standard specification, including the following optional tags [48, 56]:

- The **priority** tag indicates which pages are the most important ones.
- The **lastmod** tag gives the date of the last modification. Used along with the frequency attribute.
- The **changefreq** tag is the frequency of scanning by the robot, from **always**, for a very big Web site with pages changing continuously, to **yearly** or **never**, for static pages. As for W3C specifications of formats with a version number.

The Sitemap must [58]:

- Begin with an opening <urlset> tag and end with a closing </urlset> tag.
- Specify the namespace (protocol standard) within the <urlset> tag.
- Include a <url> entry for each URL, as a parent XML tag.
- Include a <loc> child entry for each <url> parent tag.

A sample Sitemap that contains just one URL and uses all optional tags is shown below. The optional tags are in italics [48].

```
<?xml version="1.0" encoding="UTF-8"?>
< urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
 < url>
  < loc>http://www.example.com/</loc>
  < lastmod>2005-01-01</lastmod>
  < changefreq>monthly</changefreq>
  < priority>0.8</priority>
 </url>
</urlset>
```

The HTML format is for visitors of your Web site. It may display links, titles, descriptions or other information. It is scanned by search engines and allows giving URL of pages that are not indexed, specially in the case of multi-level sub-directories, since deeper levels are not always scanned. Web sites owners are advised to put the link to the HTML sitemap on the home page [56].

An RSS file is a valid sitemap for Google and other search engines, but for recent pages only [56]. The ROR format provides several pre-defined terms for describing objects like sitemaps, products, events, reviews, jobs, classifieds, etc. The format can be extended with custom terms. RORweb.com is the official Web site of ROR; the ROR format was created by AddMe.com as a way to help search engines better understand content and meaning. Similar concepts, like Google Sitemaps and Google Base, have also been developed since the introduction of the ROR format [55].

The text format gives only the list of URL of pages to be indexed. And it is accepted by Google [56].

In 2005, the search engine Google launched the Sitemap 0.84 Protocol, which would be using the XML format [63]. Google introduced Google Sitemaps so Web developers

can publish lists of links from across their sites. The basic premise is that some sites have a large number of dynamic pages that are only available through the use of forms and user entries. The sitemap files can then be used to indicate to a Web crawler how such pages can be found [47].

Additionally, when an XML sitemap is registered, Google produces an analysis of problems encountered and a report of errors, and statistics. Google also, returns which requests to search engines will lead to the Web site pages and the pages that have not been indexed [56].

Furthermore, Sitemap 0.90 was offered under the terms of the Attribution-ShareAlike Creative Commons License and had wide adoption, including support from Google, Yahoo!, MSN, Ask and Microsoft [58].

Since MSN, Yahoo, Ask, and Google use the same protocol, having a Sitemap lets the four biggest search engines have the updated page information. Sitemaps do not guarantee all links will be crawled, and being crawled does not guarantee indexing. However, a Sitemap is still the best insurance for getting a search engine to learn about your entire site [47].

Lots of Web sites provide tools to generate different formats of Sitemaps, such as: Webmaster Tools, Simple Map Sitemap Generator, xml-sitemaps.com Sitemap Generator, Rocketface ROR/RSS Sitemap Generator, Sitemaps pal and many others. For further reading about those tools and about sitemaps we refer the reader to [23, 47, 48, 55, 56, 58, 59, and 63].

2.2 OVERVIEW OF WEB MAPPING AND CONVERSION TOOLS AND PHASES

Since many XML-based applications require bi-directional interaction with relational databases, converting back and forth between databases and XML data structures is a common requirement.

During our research, and to make the process of mapping Web data to relational database done successfully, we have been in touch with various software packages and tools that enable us to complete this process. For the completion of our work and for the reader’s benefit, we have given an overview about those tools, their benefits and main features in this section, and how they were addressed.

2.2.1 XML:WRENCH

As XML becomes more ubiquitous, many organizations are migrating to XML-based Web sites, with XML representing content and HTML presenting the content online. Separating content from presentation in this manner provides many advantages. However, much legacy content exists solely as HTML pages, and the process of converting HTML to XML is a challenge because it’s not a one-to-one conversion. To translate HTML

content to XML without losing data, developers need to convert each HTML page into three separate files: an XML instance document that contains the page content, an XSLT style-sheet with the presentation information, and a schema that represents the data content model.

XML:Wrench is a text editor for XML and similar files. XML:Wrench can be used to edit files in XML, HTML, CSS, XHTML, PHP, RSS, DTD, XSL style-sheet and other formats. Editor includes auto-suggest and auto-completion for each file-type. Each file-type has its own context sensitive help files. XML:Wrench can open and save files directly to your Web server (via FTP) [64].

XML:Wrench also includes a number of XML tools for transforming and manipulating XML/HTML files. XML:Wrench can be used to convert HTML to XML or to generate new XML/HTML from your XML source using XSL/XSLT (eXtensible Stylesheet Language Transformations) style-sheets. This can be done one file-at-a-time or using a batch command on entire directories [62]. **Figure 2.2** shows the main Interface of XML:Wrench.

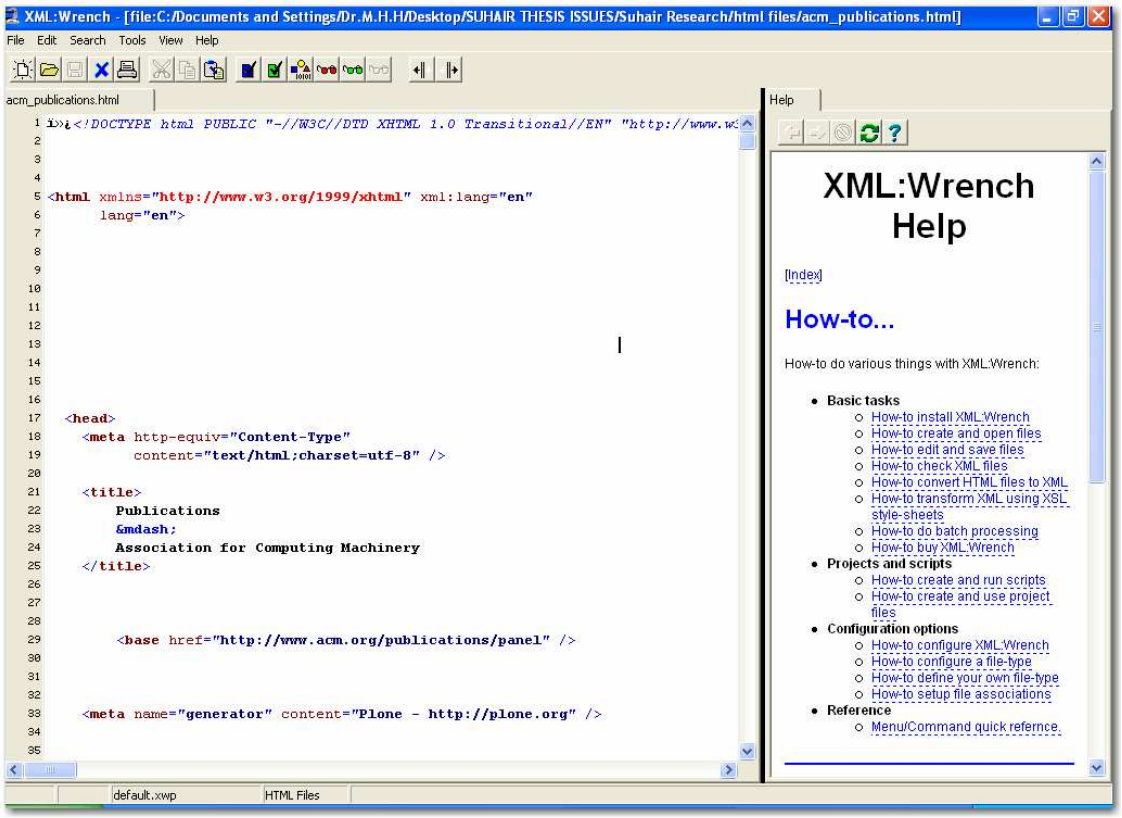


Figure 2.2: XML:Wrench Main Interface

2.2.1.1 EDITING FILES AND CHANGING THEIR TYPES

Once you have created or opened a file with XML:Wrench, you can then edit it. XML:Wrench allows multiple files to be open at a time. Files are opened onto separate 'pages' and the user can switch between them by clicking on the tabs along the top of the main edit panel.

If we want to, we can move a page over to the right hand side, click on the tab and then, holding the mouse down, drag the page over to the right. This can be useful if we wish to compare two documents, or just as a way of grouping several open files.

File-types are really a key part of how XML:Wrench works. At any given time, XML:Wrench assumes that the file in the edit window is of a given file-type. File types include, XML, HTML, CSS, plain text and much more. Users can also add one of their own if they wish.

The current file-type is used by XML:Wrench to change certain functionality. The syntax high-lighting is based on the current file-type. As is the context-sensitive help that is provided when pressing Ctrl+F1. The behavior of the preview buttons; the new file templates, auto-close and auto-completion are all determined by the current file-type.

The current file-type is set automatically when you open a file, and usually determined by the file extension. However the user can change this at any time. There are several reasons for doing this. The user might have several file-types that all use the same extension, or might also have a file containing text of several different types. For instance, users could have some CSS or PHP code nested inside some HTML statements. This HTML might then be embedded inside some XSLT elements. Depending on what part of the file the user was editing, he or she might want to change the current file-type between CSS, HTML and XSLT.

To change the file-type we select File | Change File type... from the main menu. Then select the file-type we want to use from the drop down list.

2.2.1.2 CHECKING XML FILES

It's important that XML files are syntactically correct. This is rather different from HTML, where by and large Web browsers would tolerate a measure of badly formed code.

The checks on an XML file work at two levels:

- check is well-formed
- check is valid

Firstly, a well-formed XML document is one that matches the basic specifications for any XML file. This means that all tags starting with a < must have a matching >. Further any attributes within the opening tag element must be quoted.

Also, the open and closing elements must match. When a close tag is encountered, it must hold the same text value as the most recent start tag. XML is case sensitive, so open and close tags must be in the same case.

To check a document is well-formed, we click on the well-formed button or select it from the Tools menu. If the document is well-formed we will get a short message saying OK. Otherwise XML:Wrench will display an error message and move the edit cursor to the line containing the error.

Secondly, a valid XML document is one that is well-formed and also matches criteria laid out in a Document type definition. This is typically specified in a DTD file referenced at the top of the XML file. If the XML does not contain DocType information or a DTD then there is no difference between a check for well-formed and a check for validity.

To check a document is valid, we click on the valid button or select it from the Tools menu. If the document is valid, we will get a short message saying OK. Otherwise XML:Wrench will display an error mesage and move the edit cursor to the line containing the error.

Finally, there is no need to check both well-formed and valid. If a document is valid then it must also be well-formed.

2.2.1.3 CONVERTING HTML FILES TO XML

XML:Wrench can be used to convert an existing HTML File to XML. The converter will attempt to make legal XML from the HTML without, as far as possible, changing the content. Whether the conversion is 100% successful depends on the original HTML file. After conversion, we can check the file using the XML check well-formed or validate options.

To begin a conversion, we select 'Convert HTML to XML' from the Tools menu. This will display a dialog box similar to the one shown in **Figure 2.3**.

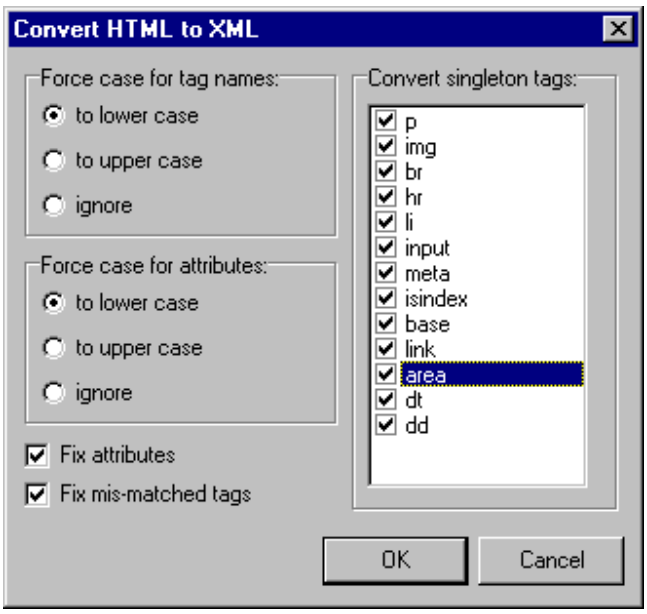


Figure 2.3: Convert HTML to XML Dialog Box

The HTML standard does not care about the case of text inside elements. The XML standard does - usually elements and attributes are required to be lower case.

Some attributes in HTML do not have any value, for example the 'selected' or 'noborder' attributes. This is not legal in XML. If we check the fix attributes check box, these will be converted automatically.

Also, HTML tags don't have to be paired up exactly. For example:

 This is italic and bold is legal in HTML but forbidden in the XML standard. In addition, many HTML pages often contain mis-matched or incomplete tags. HTML browsers have tended to be very forgiving of illegal HTML and done their best to cover up errors. If we tick the Fix mis-matched tags check box, XML:Wrench will attempt to resolve any mis-matched tags.

HTML contains a number of singleton elements that do not have content or may be used without content. These include such elements as P, IMG, BR and HR. If we select an element in the Convert singleton tags box, it will be converted to the form <x />.

To make life easier XML:Wrench keeps track of things like current folders, files, recently used files and so on. This information is saved between sessions in a default project file.

If we want to, we can create our own project files for the different projects we are working on. XML:Wrench then remembers the files we were editing, recent files used separately for each project. The project file also stores any scripts we have created.

2.2.2 OXYGEN XML EDITOR

<oXygen/> XML Editor is a cross-platform application for document development using structured mark-up languages such as XML, XSD, Relax NG, XSL, and DTD. It provides the tools for XML authoring, XML conversion, XML Schema, DTD, Relax NG and Schematron development, XPath, XSLT, XQuery debugging, SOAP and WSDL testing [61].

<oXygen/> offers developers and authors a powerful Integrated Development Environment. Based on proven Java technology, the intuitive Graphical User Interface of the <oXygen/> XML Editor is easy-to-use and provides robust functionality for editing, project management and validation of structured mark-up sources. Coupled with XSLT and FOP transformation technologies, <oXygen/> supports output to multiple target formats, including: PDF, PS, TXT, HTML and XML.

<oXygen/> is the XML Editor of choice for developers, authors and integrators that demand high-quality output with a flexible and robust, single-source, structured mark-up environment.

The integration with the XML document repositories is made through the WebDAV, Subversion and S/FTP protocols. <oXygen/> has also support to browse, manage and query native XML and relational databases.

The <oXygen/> XML editor is also available as an Eclipse IDE plugin, bringing unique XML development features to this widely used Java IDE. The main interface of <oXygen/> XML Editor is shown in **Figure 2.4**.

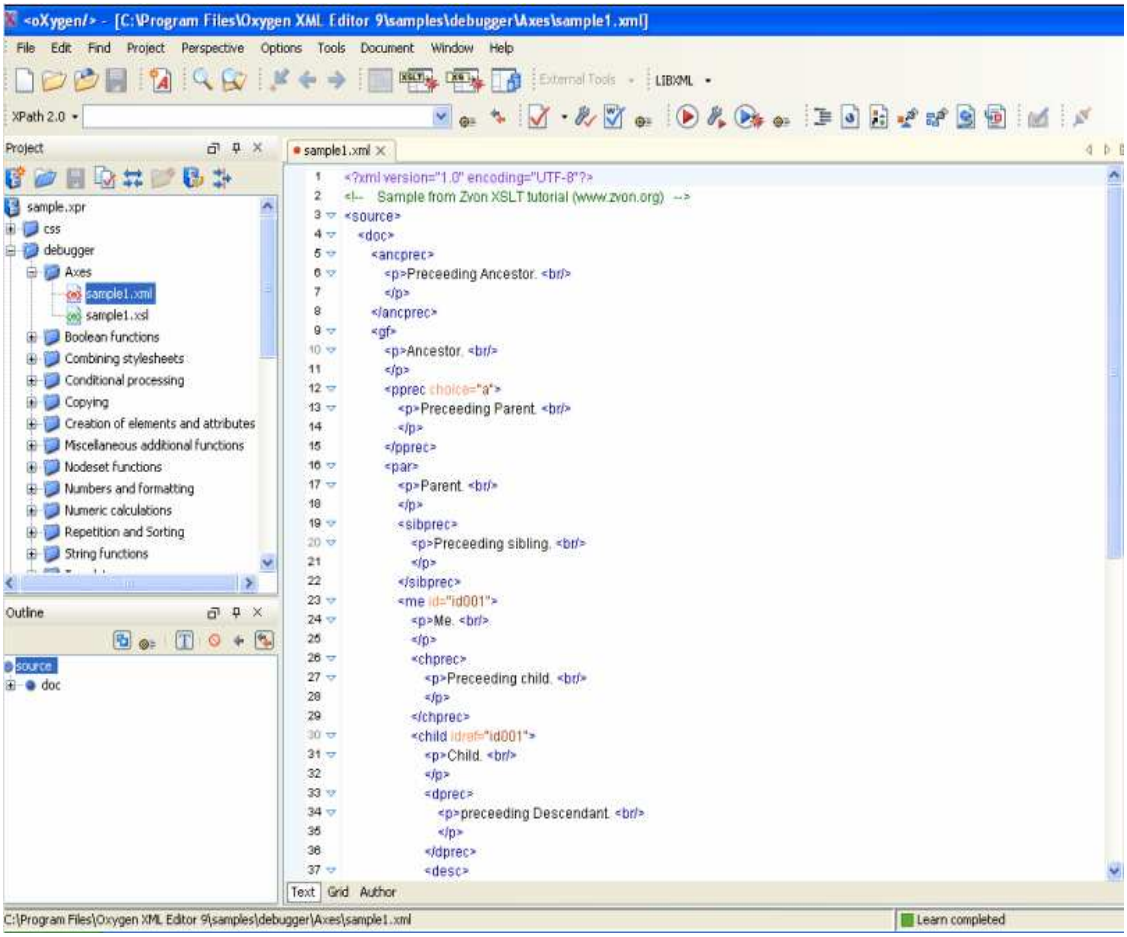


Figure 2.4: <oXygen/> XML Editor Main Interface

<oXygen/> XML editor has a large number of features regarding the editing of the XML related documents. In the subsections that follow, we are going to mention some of them. For further reading, we refer the reader to [52].

2.2.2.1 INTELLIGENT XML EDITING

<oXygen/> offers the lists of elements, attributes and attribute values through a content completion assistant. Unlike other editors that offer all the available entries, for example all the element names defined by the document XML Schema, <oXygen/> shows only those entries valid in the editing context. In this way the XML document is kept valid most of the time and the user does not have to know about the relations between elements.

In the following sample shown in **Figure 2.5**, you can see that the list of possible elements for "tgroup" element contains "collspec", "tbody" and "thead" which are exactly what the DocBook DTD has defined.

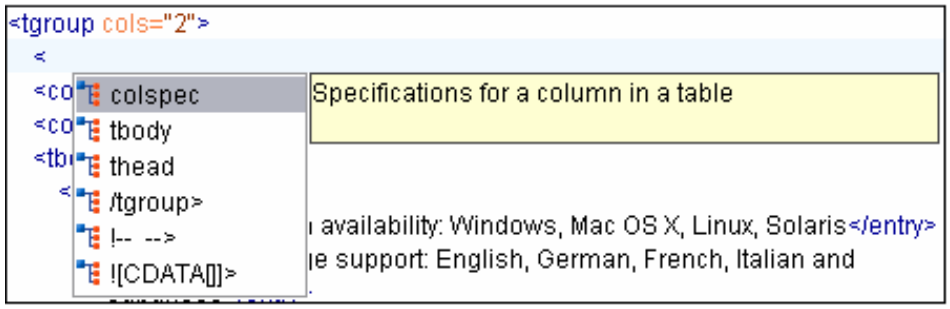


Figure 2.5: Lists of Possible Elements in Intelligent XML Editing

Also, <oXygen/> provides Support for learning words. The user need to turn this option on as it is disabled by default. All the words in the document will be indexed as they are written. When the user enter a word prefix and press CTRL+Space, all the words starting with that prefix are presented through the content completion window, as shown in **Figure 2.6**.

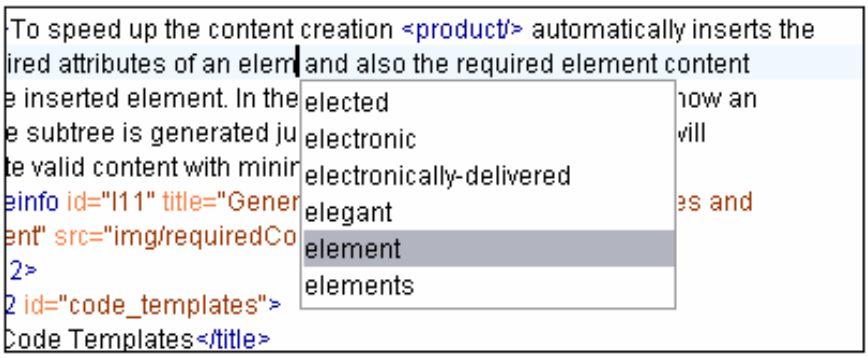


Figure 2.6: Support for Learning Words in Intelligent XML Editing

<oXygen/> also provides much more intelligent XML editing features. What we have mentioned above were just some of them.

2.2.2.2 XML SCHEMA EDITOR

The schema diagram simplifies the development and understanding of the schema files. <oXygen/> offers a side by side presentation of the schema source and diagram. The diagram is synchronized in real time with the source editor: selecting an element in the diagram highlights the corresponding element in the source editor, while moving the caret in the source editor changes the selection in the diagram view.

The schema diagram renders all the XML Schema and Relax NG components and allows the user to quickly navigate to the referred definitions of elements, attributes, types, groups, patterns, etc.

Two types of visual schema diagrams are available for a schema: The Full Model View and The Logical Model View.

- **Full Model View**

  The Full Model View provides a one-to-one correspondence between the schema components and the graphical nodes. References to different components can be expanded in place in the diagram (for instance element or attribute references, base types, or in case of Relax NG schemas, the pattern references).

- **Logical Model View**

  The Logical Model View displays a more compact diagram obtained by resolving the references, type extensions and type restrictions, redefinitions etc. for XML Schema and by applying the simplification rules for Relax NG.

A list with the defined schema components (elements, attributes, patterns, etc.) presented in the Components View along with the Outline View simplifies the navigation through large schemas. No matter the schema language, the smooth editing and the clear and suggestive rendering make schema editing more fun and easier than ever.

The Visual Schema Editor is integrated in the <oXygen/> standalone distribution and the Eclipse IDE plugin. It is activated when opening an XSD (XML Schema) or a RNG (Relax NG XML Syntax) file.

Editing actions are available on the full model view allowing the addition of new child or sibling elements on the fly so that the schema remains valid. **Figure 2.7** shows the visual schema diagram full model view.

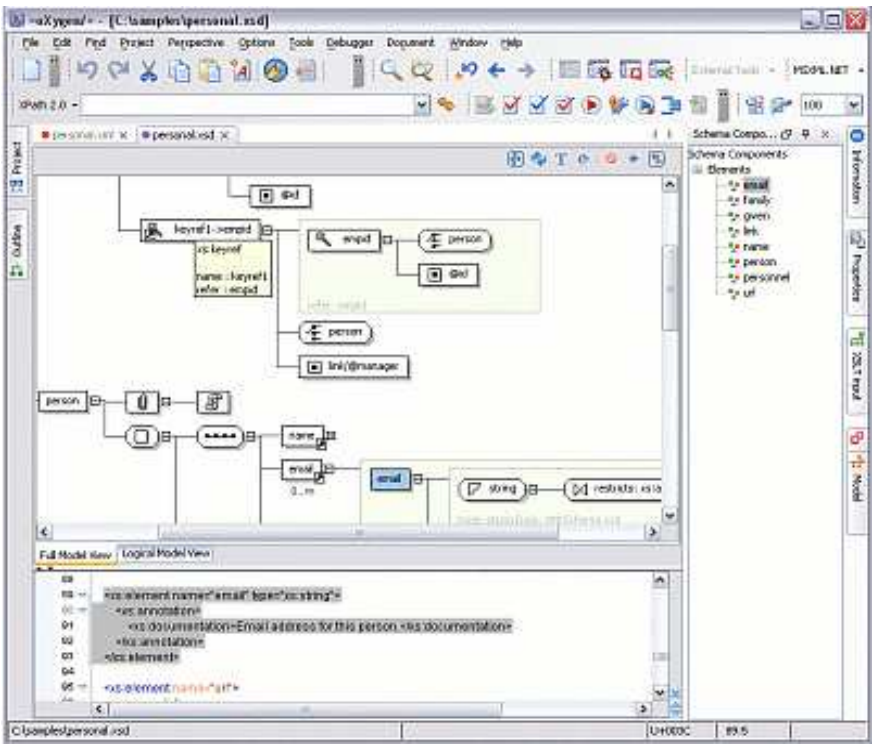


Figure 2.7: The Visual Schema Diagram Full Model View

2.2.2.3 XML AND RELATONAL DATABASES

<oXygen/> XML Editor can perform XQuery/XPath queries against a native XML database through a connection to the database server. A dedicated collection of database exploring views are grouped together in a database perspective layout. Using the database perspective users can browse tables or collections from databases, execute XQuery or SQL queries, inspect or modify data, specify XML Schemas for the XML fields and collections.

The database support includes many of the popular servers, operating either as native XML storage: Tamino, XHive, MarkLogic, TigerLogic, eXist, Berkeley or mixed, as relational and XML at the same time: DB2, SQLServer, Oracle. We have to mention that only eXist, Berkeley and JDBC (limited to table browsing) database support is included into the Oxygen Professional and Academic Editions.

The XML Database Perspective contains an explorer over the defined connections. We can define multiple connections at one time and browse them in parallel. XML Database Perspective also contains a table component for data inspection/editing and the editor area. The XML Database Perspective is shown in **Figure 2.8**.

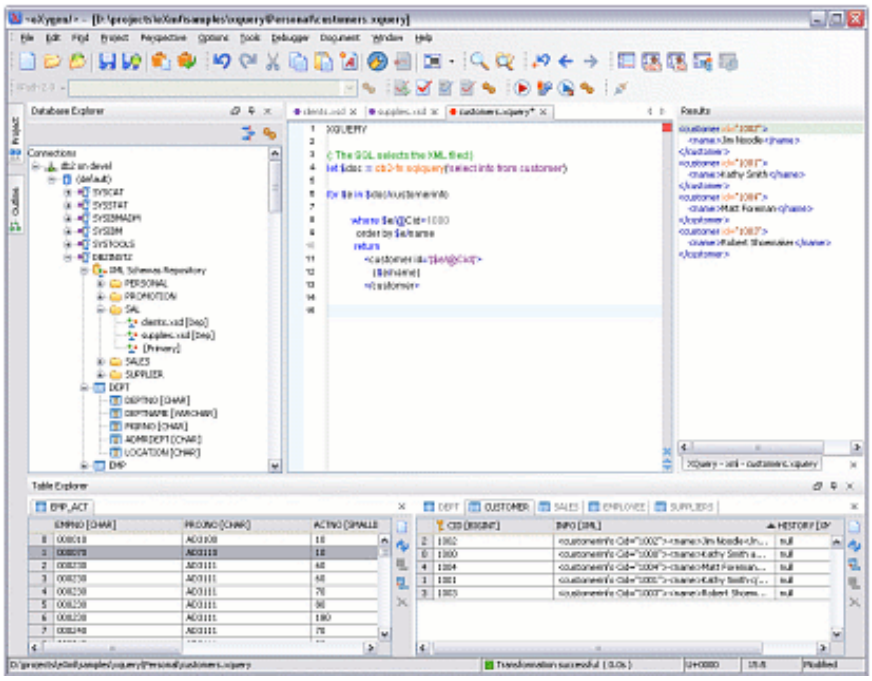


Figure 2.8: The XML Database Perspective

The database browser shown in **Figure 2.9** shows the database schemas, catalogs, tables, and field names. The user can click on a table and choose to edit the data or export it as XML.

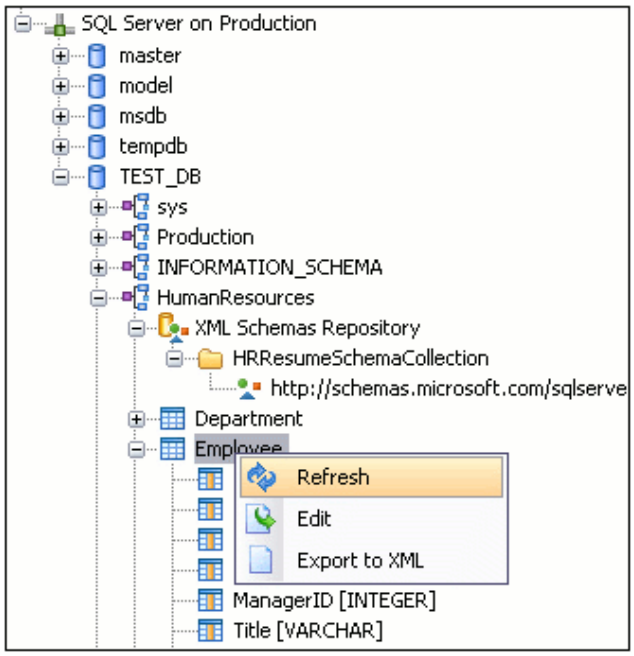


Figure 2.9: The Database Browser

If the table contains a column of the XML type, we can double-click it and open its content into the editing area as shown in **Figure 2.10**. Then, we use the normal save action to commit its new value into the database. Other operations available in the table editor are: delete, insert and duplicate rows.

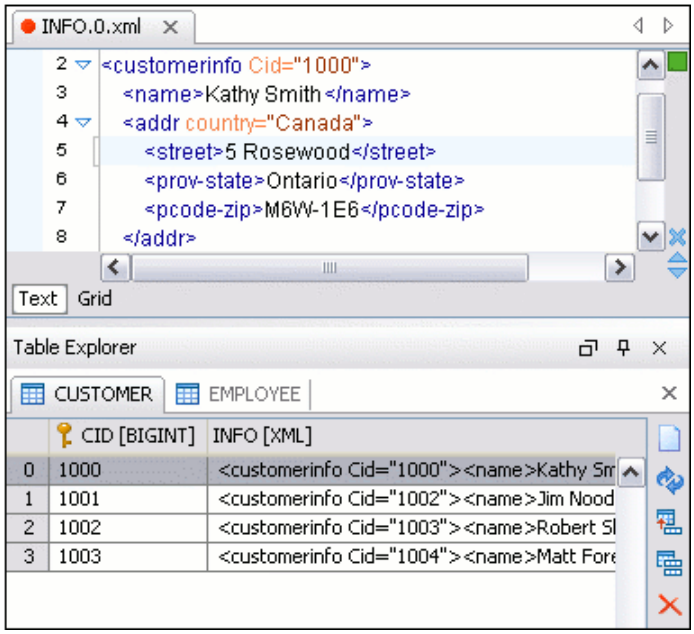


Figure 2.10: Editing Table Contents

During our research, we found that <oXygen/> XML Editor can be used to convert XML documents to XML Schemas. XML Schemas express shared vocabularies and allow machines to carry out rules made by people. They provide a means for defining the structure, content and semantics of XML documents. In more detail, XML Schema was approved as a W3C Recommendation on 2 May 2001 and a second edition incorporating many errata was published on 28 October 2004 [61].

The conversion process can be done by selecting the menu command Tools | Trang Converter. This pops up the dialog box shown in **Figure 2.11**. We select the input file type ‘XML Documents’, and add an input file which is an XML document. Then we select the output file type ‘W3C XML Schema’, and choose an output file. Finally, after the input and output is specified, we click the ‘Convert’ button and the document will be converted to an XML Schema.

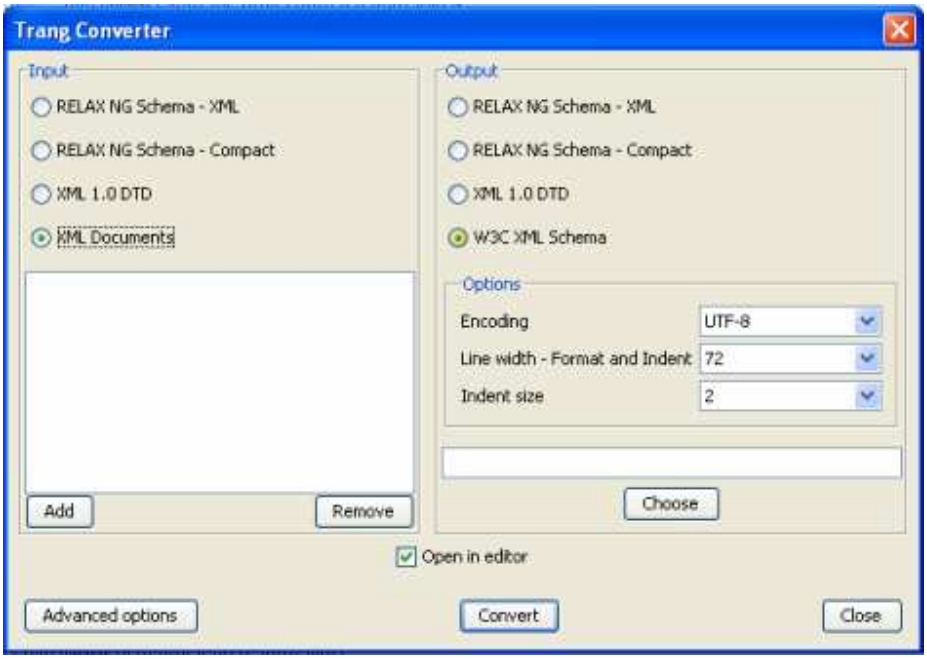


Figure 2.11: Trang Converter Dialog Box

2.2.3 ALTOVA XMLSPY 2008

Altova XMLSpy 2008 is the industry standard XML Development Environment for designing, editing and debugging enterprise-class applications involving XML, XML Schema, XSLT, XQuery, SOAP, WSDL and Web service technologies. It is the ultimate productivity enhancer for J2EE, .NET and database developers. In the following subsections, we will give an overview of this tool.

2.2.3.1 ALTOVA XMLSPY OVERVIEW AND MAIN FEATURES

XMLSpy is an integrated Development Environment (IDE) for the development of XML projects. XMLSpy can be used, among other things, to edit and process a variety of XML and other text documents; to import to and export from XML documents (including to and from databases); to convert between certain types of XML documents and other document types; to link different types of XML documents in projects; process documents with the built-in XSLT 1.0 processor, XSLT 2.0 processor and XQuery 1.0 processor, and to even generate code from XML documents [45].

The XMLSpy 2008 relational database integration functionality overcomes interoperability challenges, allowing users to easily query database and convert back and forth between databases and XML files. **Figure 2.12** represents a screen shot of XML graphical user interface, showing the Schema/ WSDL View.

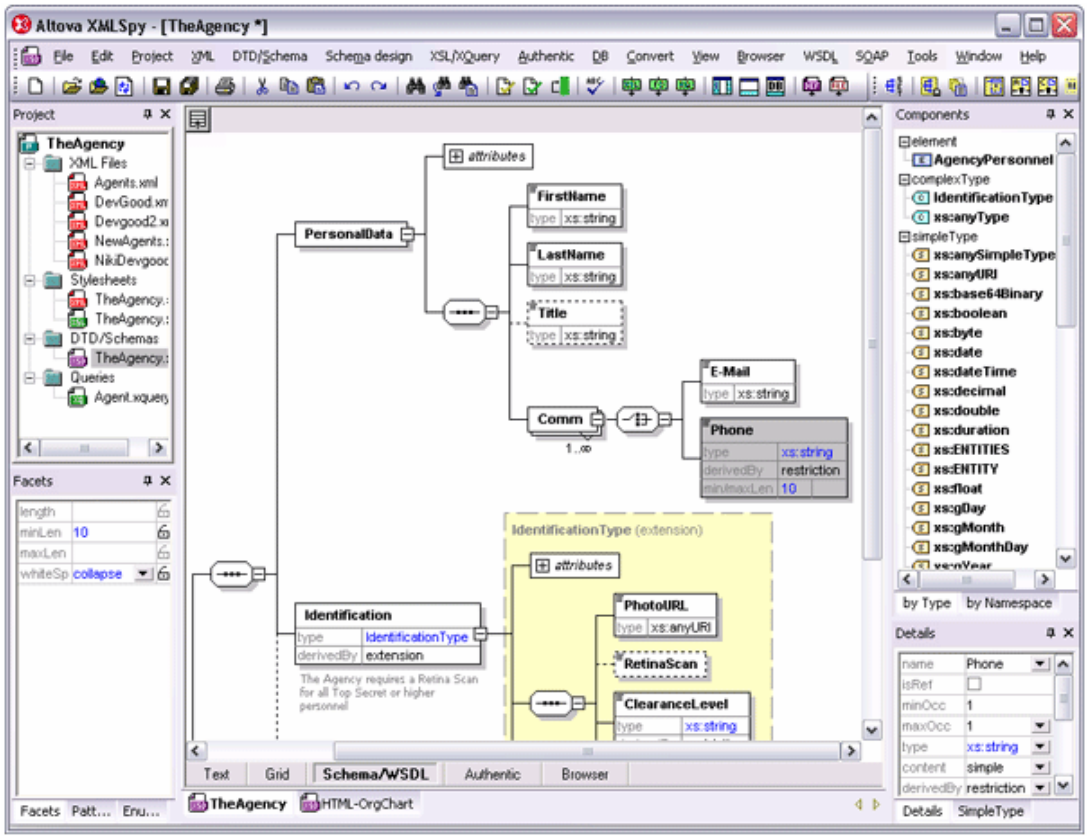


Figure 2.12: Altova XMLSpy Schema/WSDL View

To help software developers work with XML in conjunction with relational databases, XMLSpy 2008 interacts with the most popular relational databases in their native interface language, including:

- IBM DB2
- Microsoft Access
- Microsoft SQL Server
- MySQL
- Oracle
- Sybase
- Any other database supporting ADO or ODBC connectivity

XMLSpy allows users to connect to a relational database, generate an XML Schema based on a relational database, import and export data based on database structures, and generate relational database structures from XML Schemas with ease. The powerful Database Query window allows users to perform queries against the database, edit the data, and even commit changes to the database (commit currently only enabled for IBM DB2).

2.2.3.2 QUERYING A DATABASE BY XMLSPY

XMLSpy 2008 includes the Database Query window shown in **Figure 2.13,** for querying and editing database data. When you connect to a database, the Database Query window displays database tables as a hierarchical tree in the browser pane. The SQL editor tab allows you to display, edit, and execute SQL statements to query the database, either by opening existing SQL files or creating SQL scripts from scratch using drag-and-drop and auto-complete functionality.

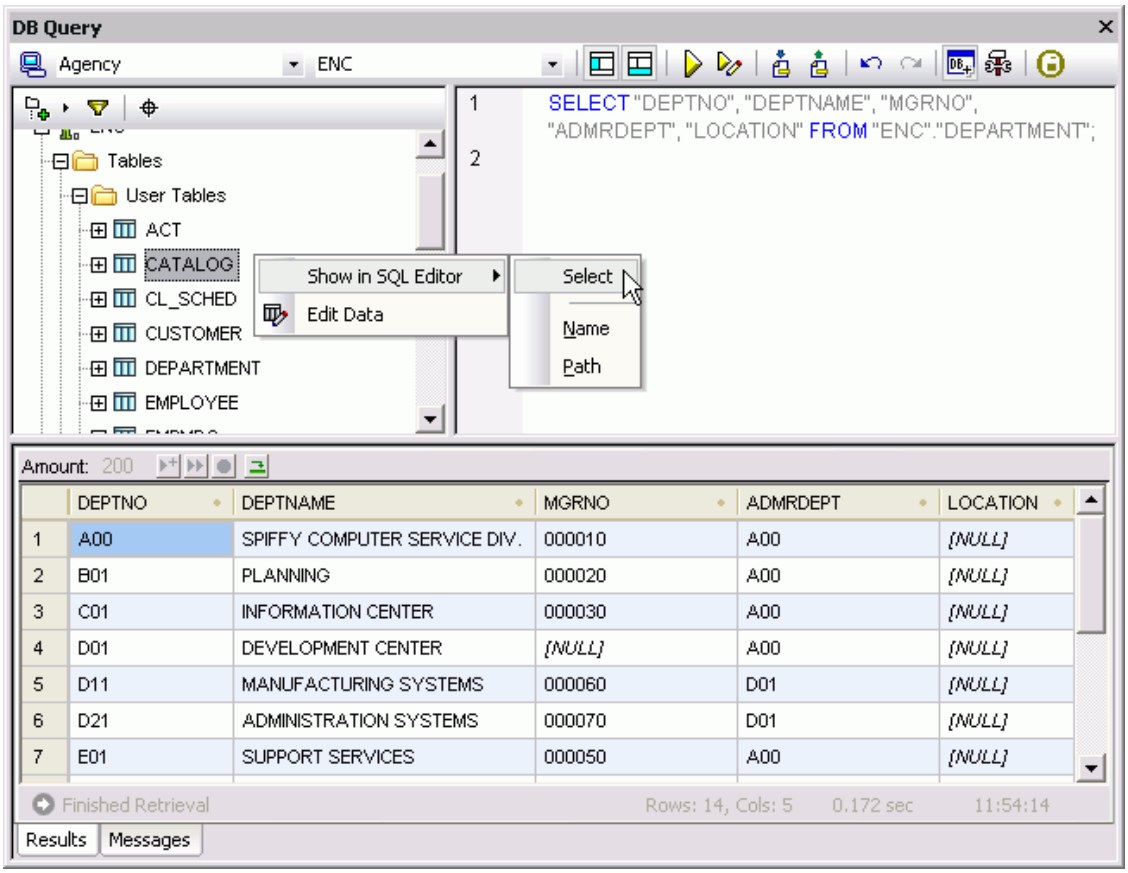


Figure 2.13: Database Query Window in XMLSpy

Once we have executed a database query, **Figure 2.14** shows how we can edit the database data directly in the results window, review changed fields (highlighted in pink), and commit the changes back to the database. Or, if the database data is XML, we can edit it in XMLSpy's text or grid view, with full validation and entry helper support, and save it back to the database, which is currently enabled for IBM DB2 9 only.

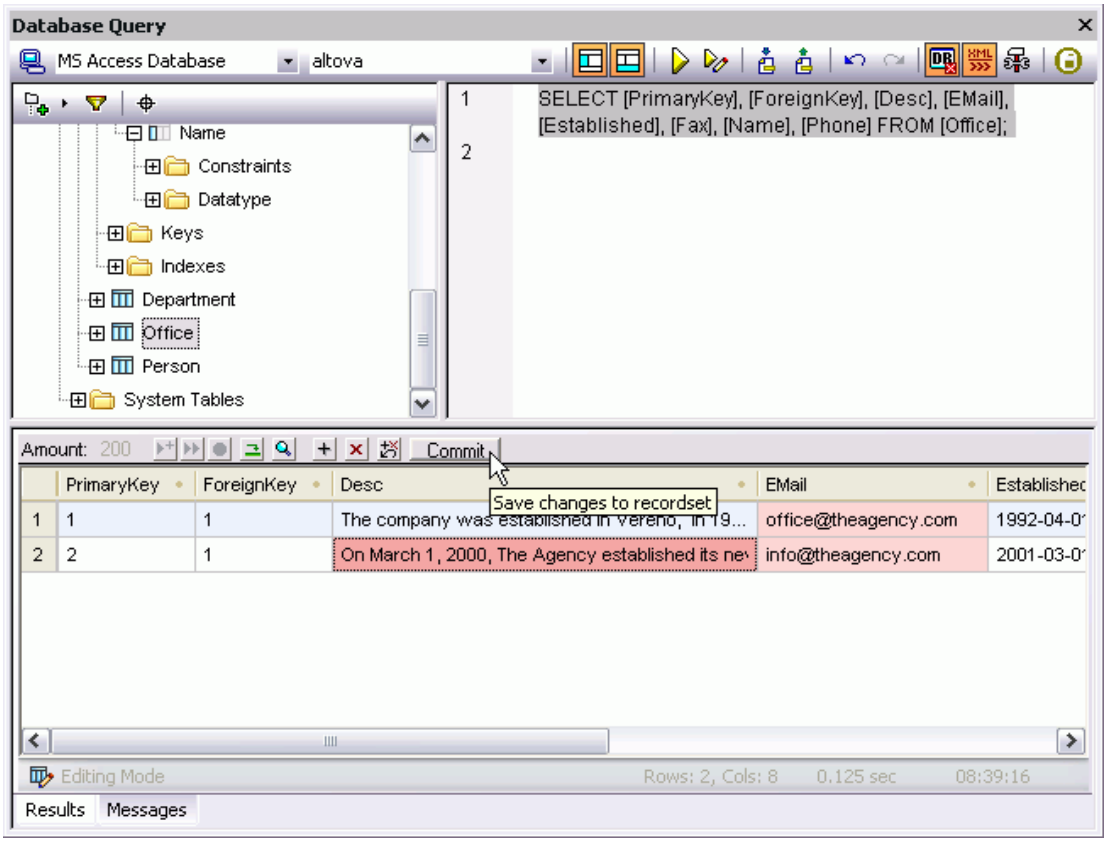


Figure 2.14: Editing Database Data in Database Query Window

The Database Query window with direct database editing support makes working with database data in XMLSpy easier and more seamless than ever.

2.2.3.3 GENERATING DB STRUCTURE FROM XML SCHEMA

The most features researchers in this field are interested in while using XMLSpy, are generating a database (DB) structure from an XML schema, exporting XML data to a database, and converting between certain types of XML documents. To complete the whole conversion process between XML files and DB structure, we must first convert the XML document; we get from either creating an XML sitemap as discussed in section 2.1 or from using the XML Wrench as discussed in section 2.2.1, to an XML schema. Then we will be able to create the DB from the schema file created.

XMLSpy allows us to create an empty database (or skeleton database) based on an existing schema file. When a DB structure is created from an XML Schema, the datatypes specific to that DB are generated from XML Schema datatypes, and vice versa if the conversion process were from a DB structure to an XML Schema, which XML Spy provides too. The conversion process could be done to several types of databases. The method described below is generally the same for each type of database.

1. We open the XML document we had, using XMLSpy.
2. And then, we select the menu command Convert | Generate DTD/Schema to open the dialog box shown in **Figure 2.15**, and then save the schema file with a suitable name.

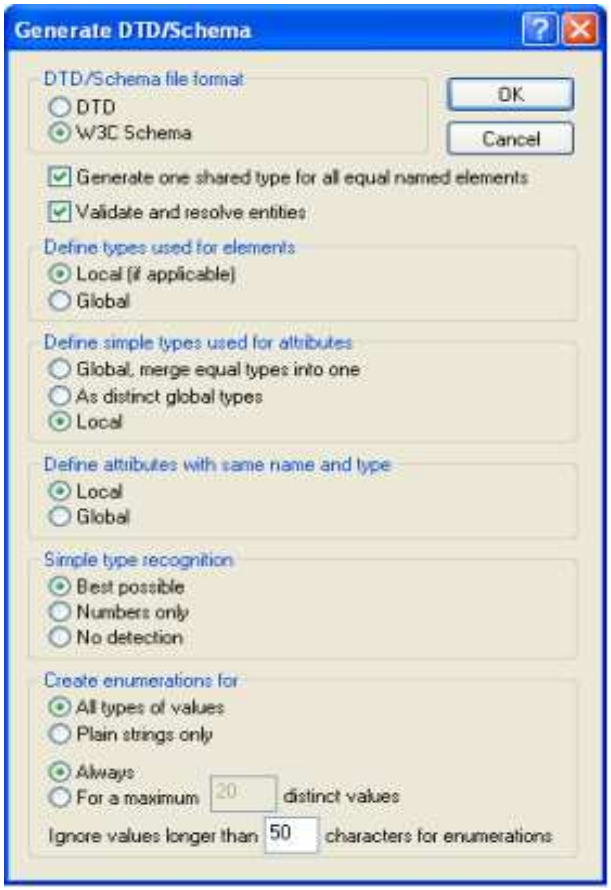


Figure 2.15: Generate DTD/Schema Dialog Box

3. Then, we open the schema file we have generated in Schema/WSDL View.
4. After that, we select the menu command Convert | Create DB Structure from XML Schema. This pops up the Connect to a Data Source dialog box shown in **Figure 2.16**, which enables us to connect to a database (DB).

Figure 2.16: Connect to a Data Source Dialog Box

5. Then we select a specific database type to connect to the required database. For example, to connect to a Microsoft Access database, we select the Microsoft Access radio button, and continue the process to select a database. We can use an existing database or create a new database in which the schema structure will be contained.
6. In the Create DB Structure from XML Schema dialog, tables are created from the schema and displayed in a tree format at the location where they will occur in the DB. For example, in the screenshot shown in **Figure 2.17**, the Address table is created and selected for export. Tables that should not be exported should be deselected (by unchecking the check box or selecting the appropriate item from the context menu for that table).

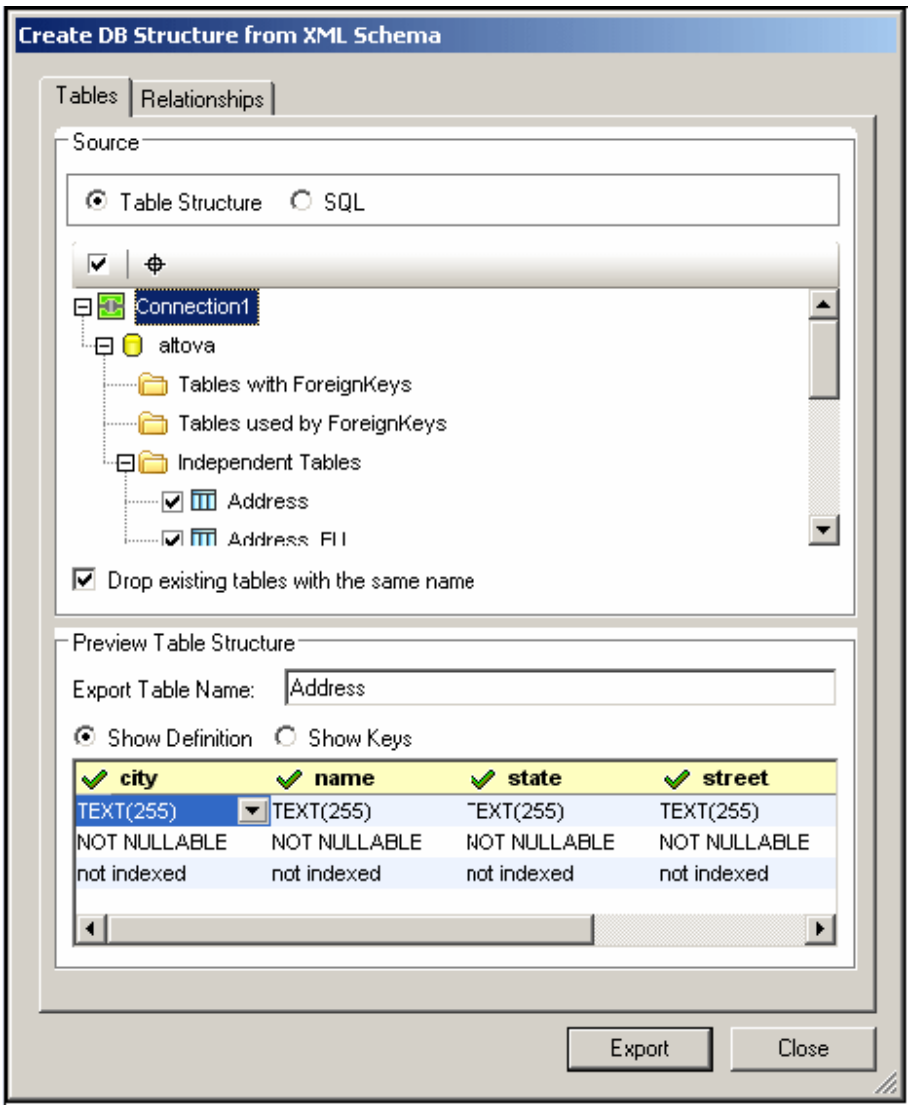


Figure 2.17: Create DB Structure from XML Schema Dialog Box

7. If we want to drop (delete) tables in an existing DB that have the same name as tables coming in from the schema, then we could check the Drop Existing Tables with the same name check box.

2.2.3.4 CREATING DB TABLES WITH RELATIONSHIPS

If the XML Schema from which the DB structure is generated has relationships defined in the form of identity constraints, then these relationships are automatically created in the generated DB structure and displayed in the Table Structure. Tables with relationships are listed under the sections: Tables with ForeignKeys and Tables used by ForeignKeys. Tables without relationships are listed in the Independent Tables section.

In the Relationships tab, we can create and modify table relationships. The tab lists all possible primary-key/foreign-key relationships as shown in the screenshot in **Figure 2.18**.

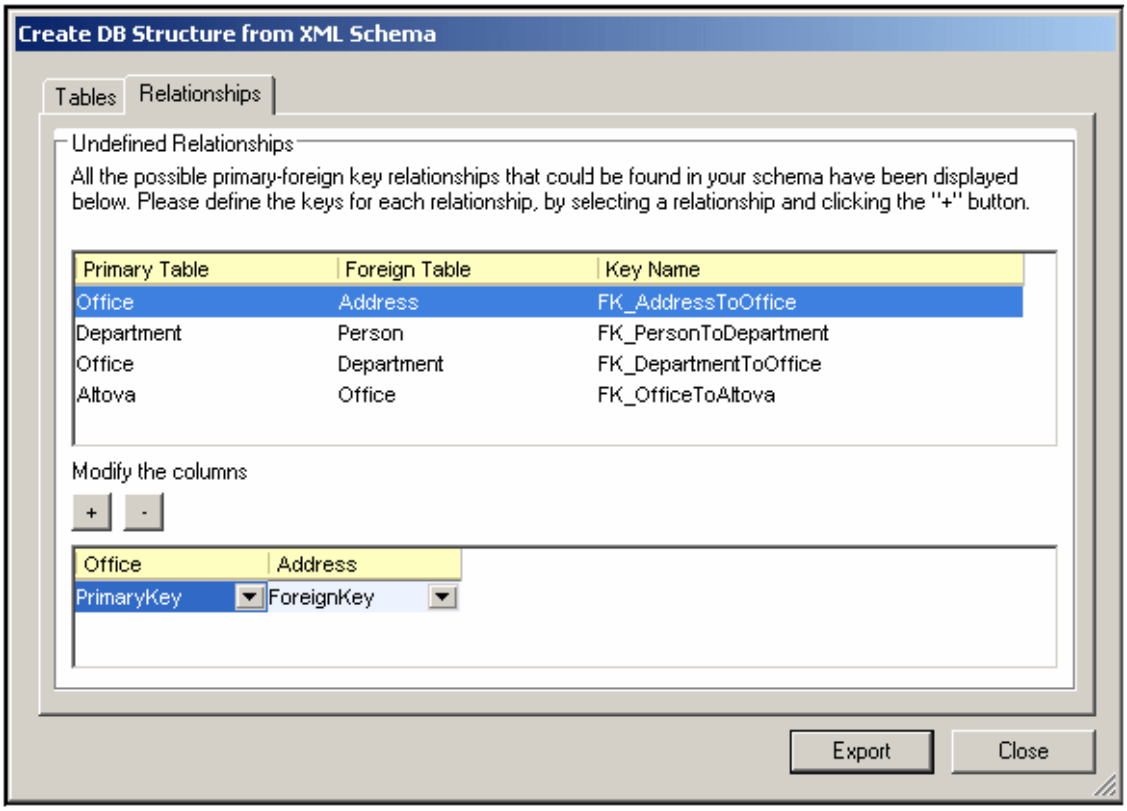


Figure 2.18: Create DB Tables with Relationships

To create a relationship, we can do the following:

1. We select one of the possible primary-key/foreign-key relationships.
2. In the lower pane of the dialog, we click the Plus button to create a relationship.
3. Then, we select the required columns in each of the two tables from the respective dropdown lists.

We can also remove a relationship by selecting it and then clicking the Minus button.

2.2.3.5 EXPORTING XML DATA TO A DATABASE

After completing the steps described in the previous sections, our database is now created, but its tables are still empty. To export the XML data to the database, we select the menu command Convert | Export to a database to open the Export to Database dialog box shown in **Figure 2.19** which allows us to specify where to start, how to handle export fields, and which elements to include. Then we just select which database type we wish

to append or create with our data, and the data is instantly converted and stored in the database.

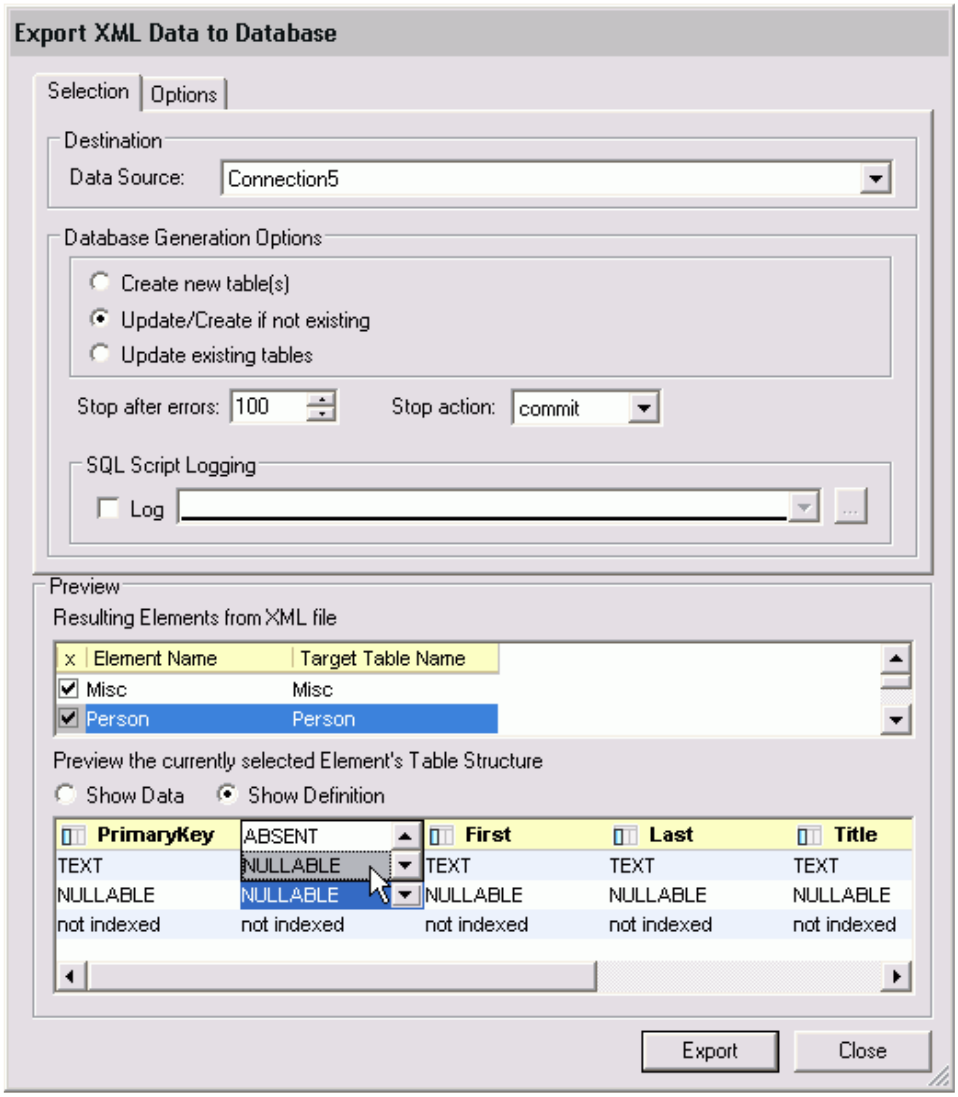


Figure 2.19: Export XML Data to Database Dialog Box

The following structure shown in **Figure 2.20** summarizes the steps of extraction and conversion that we have described above to convert Web content to a relational database.

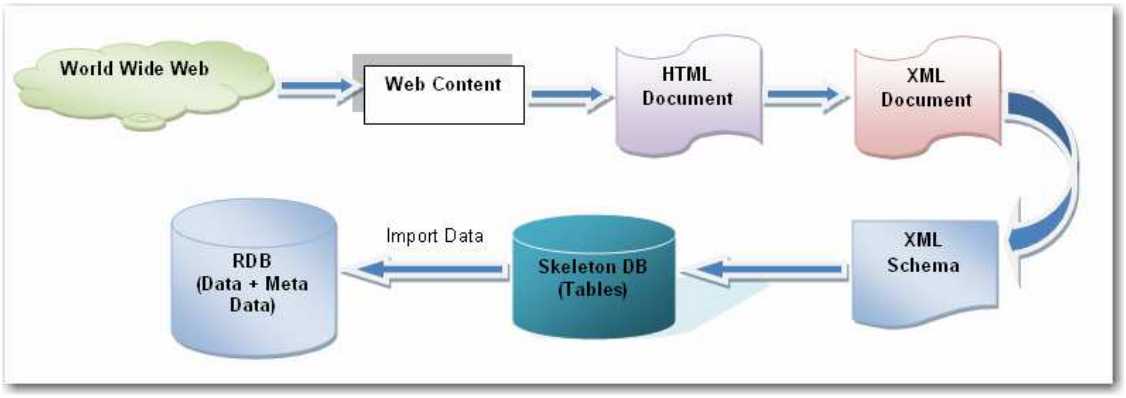


Figure 2.20: Steps of Converting Web Content to RDB Using Mapping Tools

Finally, the whole conversion process using the tools mentioned above, have been experimented to convert an XML document to an MS Access database and to an Oracle database. But even after completing exporting the XML data to the database, the tables and the data we have got weren't valuable as the data we were seeking to get from the Web sites we have analyzed and studied. So that, we couldn't build the search interface on the databases that were created. As a result, we continued our research and found out another way to create our desired database which will be discussed in details in Chapter 3.

2.2.4 OTHER CONVERSION AND MAPPING TOOLS

There are many other tools and software packages that provide mapping facilities between XML and relational databases. In this section, we are going to talk briefly about some of them and mention others that readers may be interested in.

2.2.4.1 ALTOVA MAPFORCE 2008

Altova MapForce 2008 is the premier graphical data mapping, conversion, and integration tool. The data mapping tool maps between any combination of XML, database, flat file, EDI, and/or Web service, then transforms data instantly or auto-generates a royalty-free data integration application for execution of recurrent conversions [43]. The main interface of MapForce is shown in **Figure 2.21**.

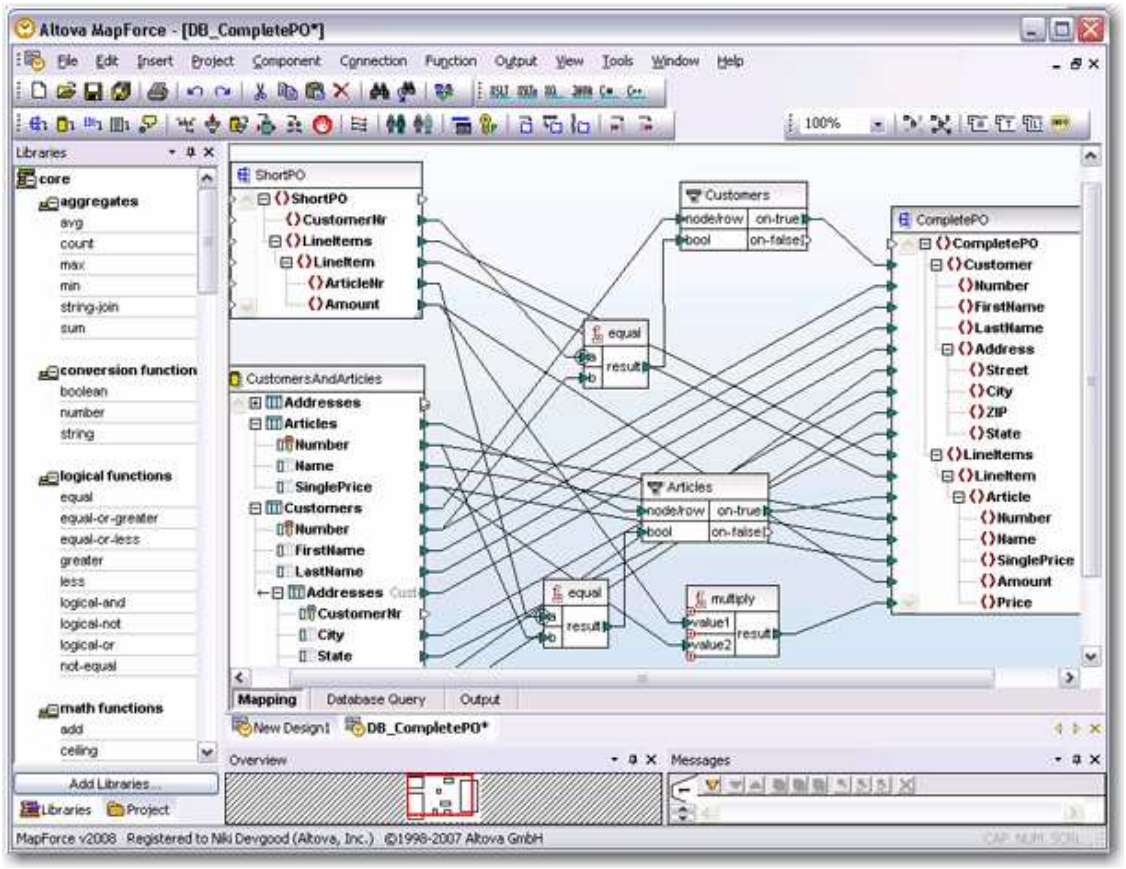


Figure 2.21: Altova MapForce Main Interface

Altova MapForce 2008 includes powerful support for database mapping, allowing users to map and convert between database data and XML, flat files, EDI, Web services and other database formats. In addition, support for building new Web services allows users to map between database data and Web services operations, and then generate code to implement the service server-side.

MapForce 2008 is a general purpose data integration and Web services implementation tool to support all major relational databases. Using MapForce, users can create graphical data mappings by dragging connecting lines between source data, data processing functions and filters, and target data structures. Database support in MapForce allows users to:

- Connect to and query all major databases.
- Convert data from any supported database type to any other.
- Map data from XML, flat files, EDI, and/or Web services into databases.
- Map relational database data into XML, flat files, EDI, and/or Web services.
- Map XML data in databases into XML, flat files, EDI, and/or Web services (IBM DB2 9 only).
- Create Web services that consume or write to databases.

MapForce 2008 supports databases as both the source and/or target of any mapping. It supports the following databases:

- IBM DB2
- Microsoft Access
- Microsoft SQL Server
- MySQL
- Oracle
- Sybase
- Any ADO/ODBC database

2.2.4.2 ALTOVA STYLEVISION 2008

Altova StyleVision 2008 lets users convert HTML to XML easily, using a visual interface. Instead of hand-coding a separate XML instance document, XSLT stylesheet, and schema, users simply highlight content from an existing HTML file, drag it across the screen, and place it in a content model pane. StyleVision 2008 converts the data behind the scenes and presents users with standards-conformant XML, XSLT, and schema (XML Schema or DTD) documents [44].

In addition, once users have converted HTML to XML, StyleVision 2008 automatically provides them with the corresponding XSLT 1.0 or 2.0 and XSL:FO (eXtensible Stylesheet Language Formatting Objects) to output their content in Word/RTF and PDF, respectively.

By design, XML separates content from presentation. XML itself is only responsible for representing data in a text format that is readable by all applications. Related technologies such as XSLT and XSL:FO are used to specify how the data in an XML document should be styled and laid out in presentation media such as HTML or PDF.

Since business data must often be published in several different media to meet the needs of customers, partners, and internal audiences, multiple style-sheets are required to present each XML document. Coding even the simplest transformation style-sheets by hand can be a difficult task, and writing multiple style-sheets to present an XML document in popular formats like HTML, PDF, and Word / RTF can become arduous and error-prone very quickly.

Altova StyleVision 2008, however, takes the headache out of publishing XML in multiple formats by auto generating XSLT 1.0, XSLT 2.0, and XSL:FO style-sheets based on a single design that users can create using drag and drop functionality. Their ONE design simultaneously transforms their content into HTML, PDF, and Word/RTF and allows them to save the corresponding standards-conformant style-sheets.

2.2.4.3 STYLUS STUDIO2008 XML ENTERPRISE SUITE

Stylus Studio 2008 XML Enterprise Suite provides a comprehensive set of XML tools and features for working with XML, XQuery, Web services, XML publishing, and many other XML technologies. It is an XML development tool to support visual XQuery editing, XQuery mapping, and XQuery debugging. And also, it provides comprehensive XSLT development support, including XSLT debugging, XSLT mapping, XSLT profiling, visual HTML-to-XSLT style-sheet design, XSL: FO, and others [60].

Stylus Studio 2008 XML Enterprise Suite includes numerous synchronized, visual XML editing views, Sense:X (Intelligent XML Editing), Integrated XML Validator, XML differencing, and much more. Furthermore, it includes a visual DTD editor, integrated DTD validator, and various DTD generation utilities.

With Stylus Studio 2008 XML Enterprise Suite, users can map data in one format to other using visual drag-and-drop mappers. Stylus Studio's mappers include support for mapping to and from XML documents, Web service data, relational data, flat files, and many more.

This suite also includes Convert to XML, a visual tool for extracting XML data from any file format including CSV, tab separated, binary data files, EDI, or any other flat format, as well as many other data import/export utilities for RDBMS, XML, and others. It also provides a powerful database and XML tools that are really useful for database architects, database developers, and Database Administrators (DBA's).

2.2.4.4 ALLORA FROM HIT SOFTWARE

Allora leverages leading edge XML mapping and database technology to give application developers bi-directional access to relational databases without the need for complex SQL or XSLT programming. Allora speeds development and deployment by accurately and transparently transforming data structures between XML elements/attributes and relational database structures [49].

Once Allora mapping is in place, full bi-directional XML-RDB access is enabled. Allora XML database transformation gives developers a simplified, consistent XML interface to relational data and includes Web Services. Whether they export data from a database into XML, import data from XML to a database, insert XML into a database, or transform or convert XML into a database, Allora and HiT Software XML Utilities could be a great help for all users in doing their job.

Allora can generate XML from any database and write XML element and attribute values into any database. It is an XML database mapping platform that consists of:

1) Design-time tools for mapping XML to relational data or text.
2) Wizard support for popular application servers and IDE tools.
3) A rich set of application programming interfaces to the Allora run-time engine.

4) A Workflow Manager tool to organize and execute a set of transformation tasks with associated XML file manipulation.
5) A set of SOAP interfaces to the Allora Web service engine Web for building distributed applications.
6) And a database connector.

Allora works with any relational database that has a JDBC or ODBC connector. Allora has been certified to work well with over 20 different databases. Allora bi-directionally transforms XML and data stored in relational databases such as DB2, Oracle, Microsoft SQL Server, MySQL, Informix, Ingres, IDMS, Datacom, ANTs, MaxDB, and Sybase.

Allora's design-time components for data transformation include a graphical Mapper application, source code wizards, object interfaces, and sample source code. The design-time graphical Mapper application lets developers simultaneously see DTDs or XML schemas and database catalog structures. Also, the Mapper can automatically create XML schemas from relational databases and vice versa. Using the mapping files generated by the Mapper, developers can request data specifying XML structures rather than database structures or SQL. These mapping files are passed to Allora's interfaces by calling applications.

Allora offers a rich, flexible set of interfaces. Developers can refer to data by XML element/attribute references, including XPATH, or by RDB catalog structures. Similarly, they can exercise XML element/sub-element methods or specify record/field SQL commands. The Allora OVLT API optimizes import and export of data within very large tables and XML documents. This API increases performance by an order of magnitude while minimizing local and database server resource requirements. Allora manages all SQL middleware connection processes for efficient and proper database interaction and data integration.

Allora's data binding object interfaces enable developers to create objects that represent database records. Data binding includes methods to iterate through records sequentially, yet retaining the XML formatting structures.

For maximum data access flexibility, Allora can also be implemented as a full-function SOAP-based Web service for remote applications. Allora Web service supports two client access modes: a SOAP interface mode and a higher level remote API mode. **Figure 2.22** presents the Allora XML database server software.

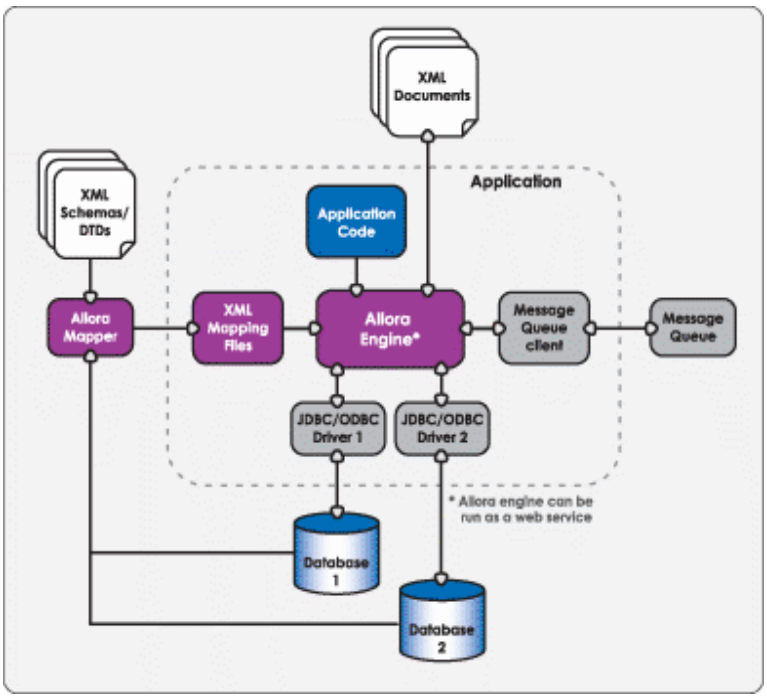


Figure 2.22: Allora XML Database Server Software

The mapping tools and software packages that we have described earlier in this chapter are just part of other existing tools related to the subject of our thesis. We feel that the tools we have selected to describe are enough for the benefits of the readers, even though interested readers may look for more.

CHAPTER 3

CONVERTING WEB DATA TO RELATIONAL DATABASES

In this Chapter, we have explained and discussed the first phase of developing GMBI, which presents how the extraction of Web data was done and then how the extracted data was converted to relational database. The whole architecture of GMBI is presented in **Figure 3.1**. And the process of extracting Web data and converting it to RDB is shown in **Figure 3.2**.

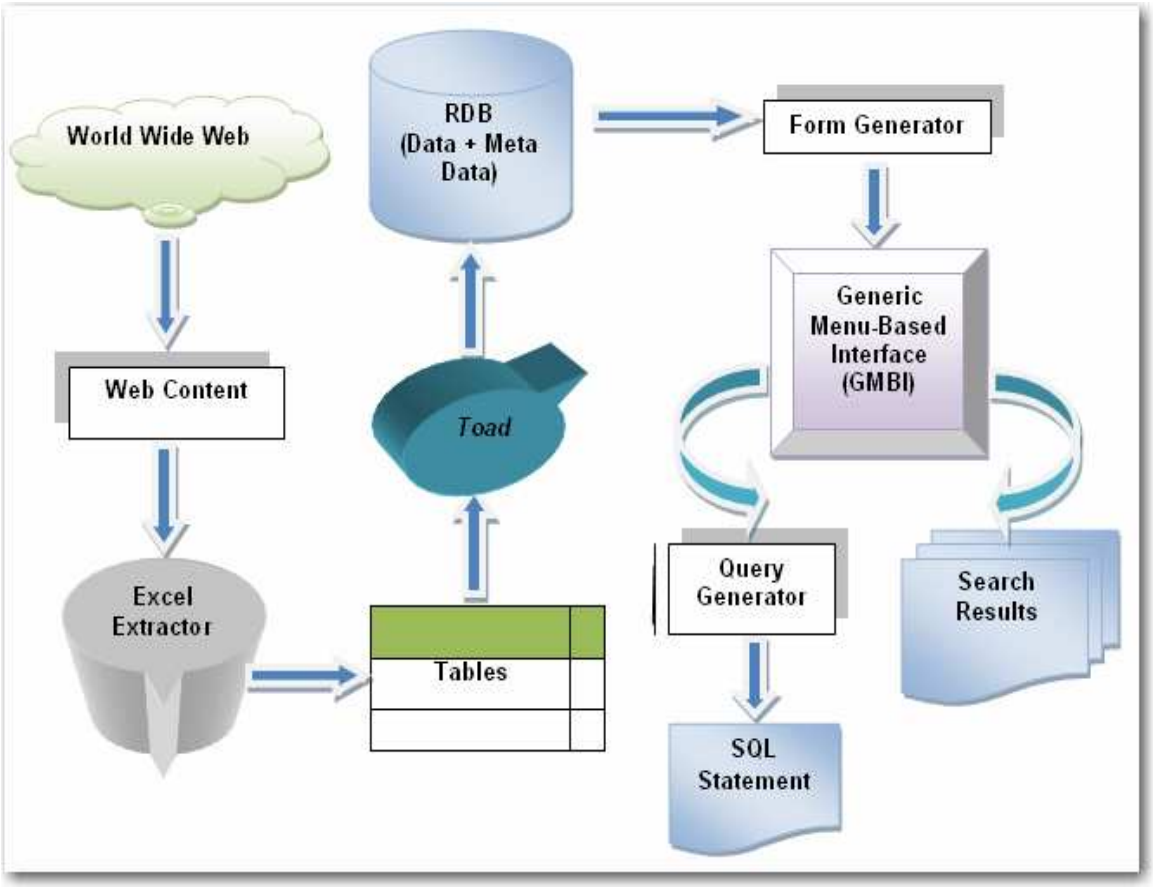


Figure 3.1: The Architecture of GMBI

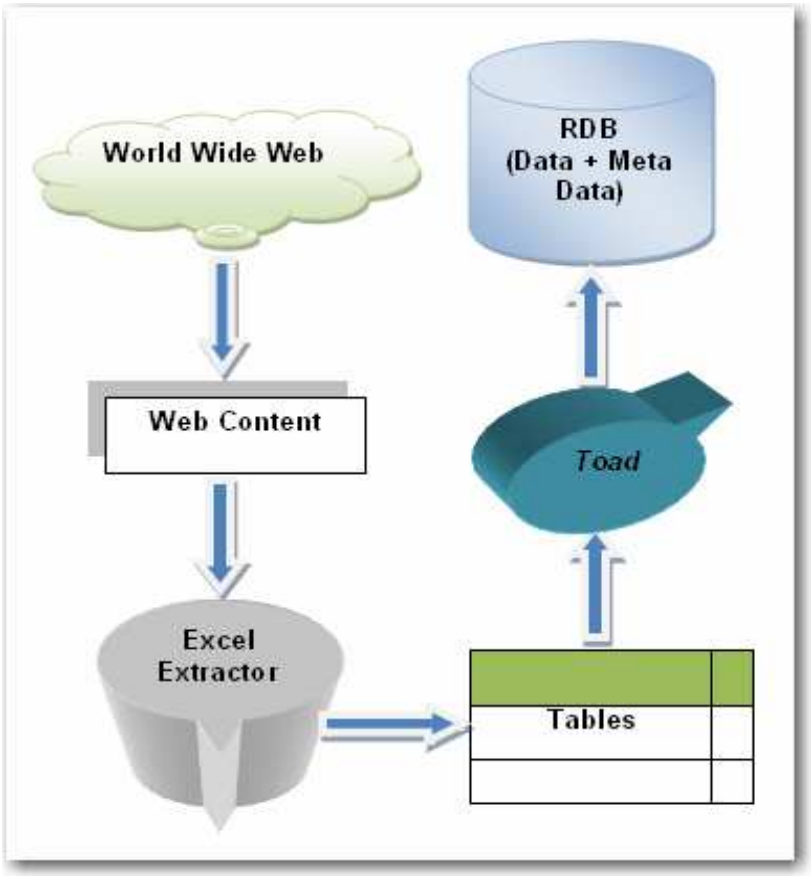


Figure 3.2: Extracting Web Data and Converting it to RDB Process

3.1 SELECTING A SPECIFIC DOMAIN

In our work, we thought that selecting a specific domain that users may be most interested in would be valuable and make the work more quantified. We have chosen to focus on Web pages that contain publications in its variety, such as books, articles and citations, journals and magazines, newsletters, conferences, educational courses, and much more.

In our domain of interest, we selected some specific Web sites to work with and try our search form on their data, specially: **www.acm.org**, **www.ieee.com** and **http://citeseer.ist.psu.edu**. From ACM, we focused on ACM Books, Journals and Magazines, Affiliated Organizations, Newsletters, Proceedings, Transactions, and Special Interest Groups (SIGs). And from IEEE, our focus and study was on IEEE Conferences, Journals, Magazines, and Educational Courses. Finally, from Citeseer, we focused on CiteSeer Articles and Citations.

We have created our database from data in those Web sites mentioned above and their publications, as will be shown in the coming sections in this chapter. And then we tested

the search form we have built and make queries on the database we have created successfully as will be discussed in chapter 4. Although our focus was on those specific Web sites, our work is generic and has the flexibility to work with most Web sites with publications.

3.2 IMPORTING WEB DATA INTO MICROSOFT EXCEL WITH WEB QUERIES

Web queries offer a handy way to import data from selected tables at a Web page into a worksheet. An Excel Web query allows us to pull data from a Web site into an Excel worksheet. It will find any tables on the Web page and let us select the ones containing data we want to put into our worksheet, allowing for dynamic updates from the Web page.

Web queries are not just useful for pulling information from standard HTML pages, but also they can be used quite nicely in situations where a standard Object Database Connection (ODBC) would be difficult or impossible to create or maintain.

3.2.1 CREATING THE WEB QUERY

To make the importing process obvious, we are going to start with a simple Web query using one of Yahoo! Web pages, finance historical stock prices, to show how this step was done. This is a great example because the data we are interested in is presented in a plain, tabular format and has little confusing information in it. But we must notice that this Web query doesn't put important information in images or through links.

We created the Web query, through the following steps:

1. We select the first cell in the excel sheet in which we want results to appear.
2. Then we choose Data | Import External Data | New Web Query to open the dialog box shown in **Figure 3.3**. And then entered the URL to query in the address area and clicked the Go button as shown in **Figure 3.4**.

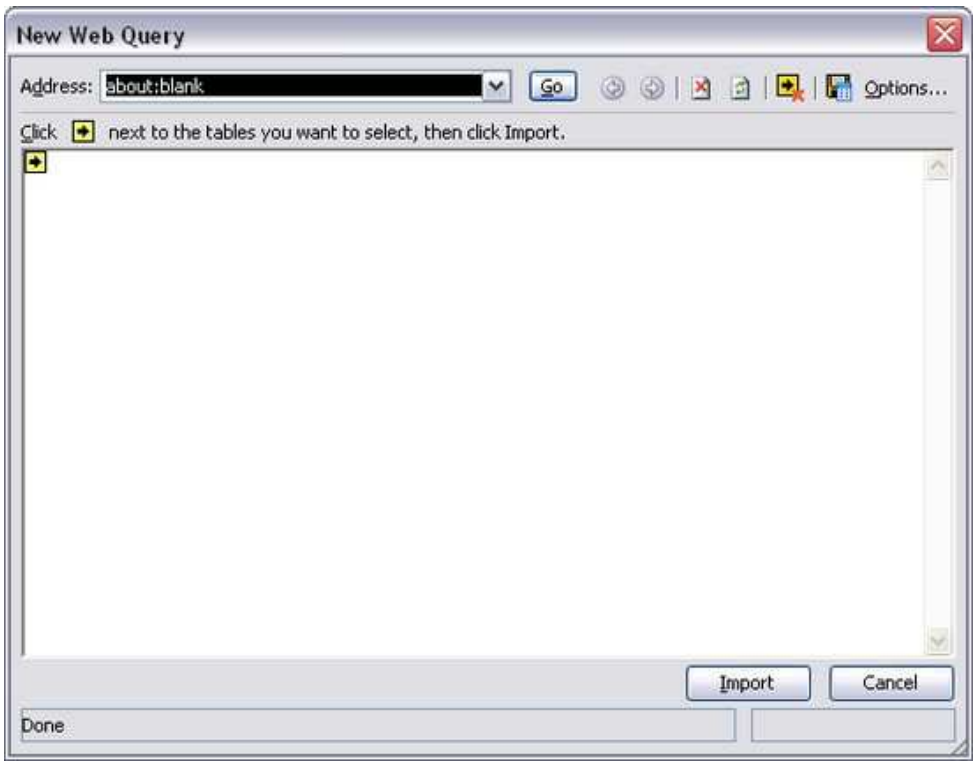


Figure 3.3: New Web Query Window

Figure 3.4: The Web Page of Interest

3. Another way to accomplish what is done in 2, is to choose Data | Import External Data | Import Data, and then open a saved HTML Document (which is the web page we want to pull data from) as shown in **Figure 3.5**.

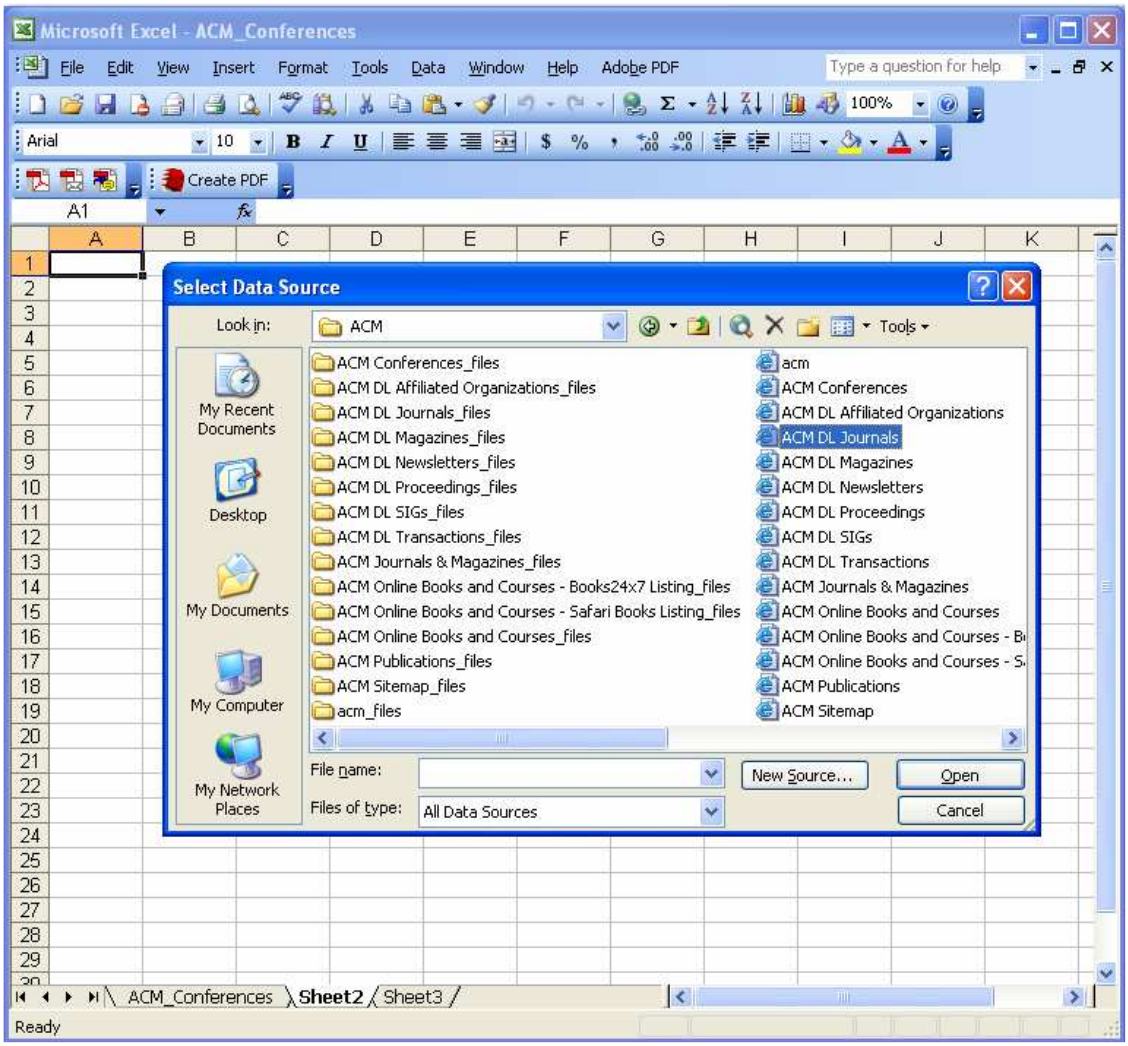


Figure 3.5: Choosing an HTML document to open the Web page

4. Select the table we want to use for the query as shown in **Figure 3.6**. And if a well-formatted table wasn't available, we could select any data we want from the Web page and then modify it to a well-formatted table, as will be discussed later in section 3.3.

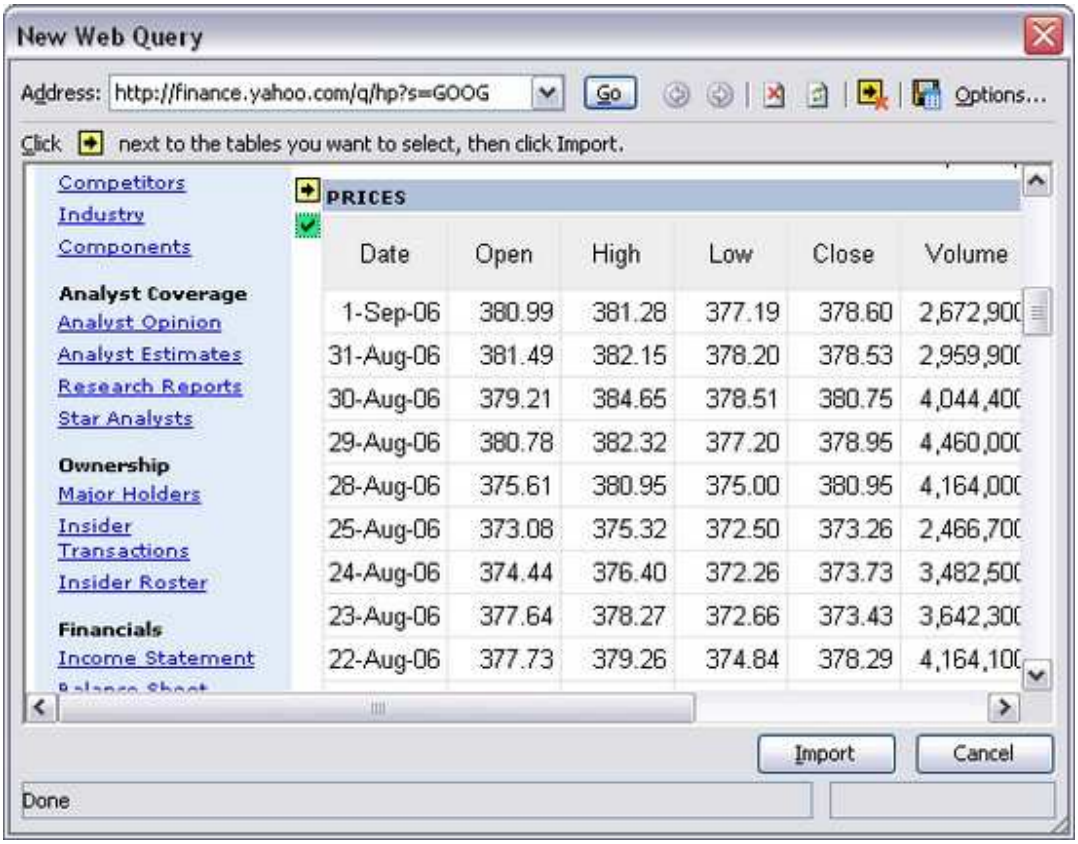
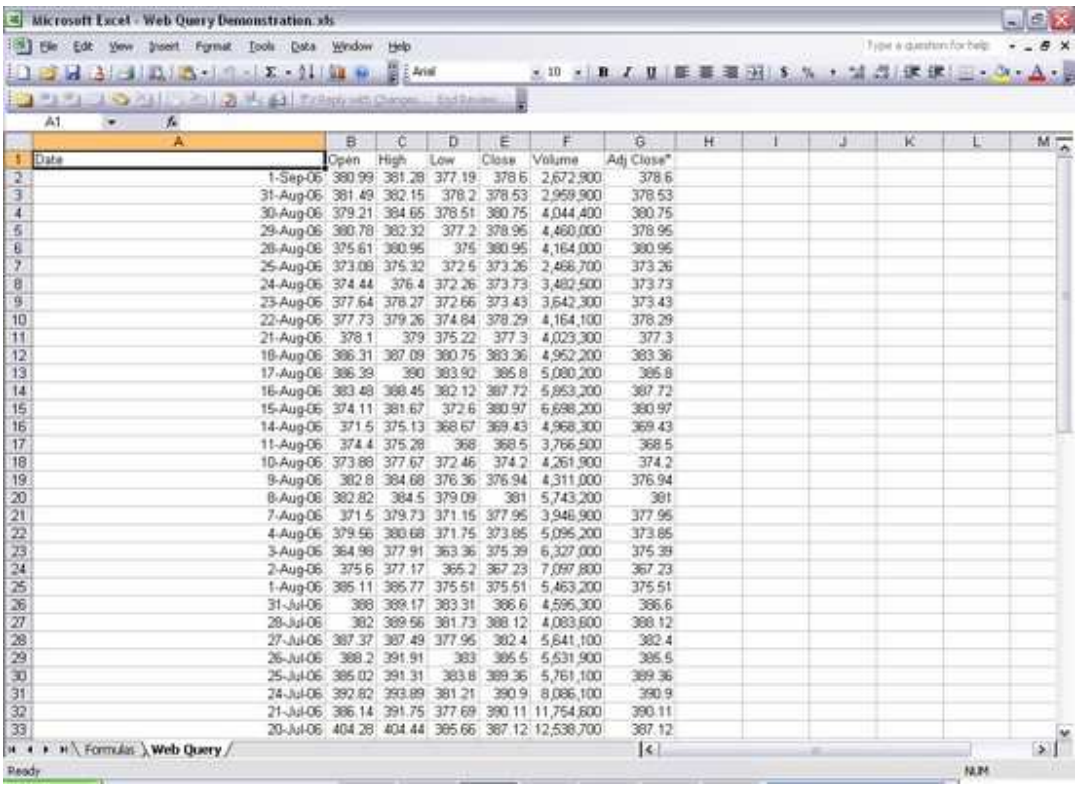


Figure 3.6: Selecting a Table from the Web Page

5. Finally, we click the Import button. And the data is now in our worksheet, as shown in **Figure 3.7**.



	A	B	C	D	E	F	G
1	Date	Open	High	Low	Close	Volume	Adj Close*
2	1-Sep-06	380.99	381.28	377.19	378.6	2,672,900	378.6
3	31-Aug-06	381.49	382.15	378.2	378.53	2,959,900	378.53
4	30-Aug-06	379.21	384.65	378.51	380.75	4,044,400	380.75
5	29-Aug-06	380.78	382.32	377.2	378.95	4,460,000	378.95
6	28-Aug-06	375.61	380.95	375	380.95	4,164,000	380.95
7	25-Aug-06	373.08	375.32	372.6	373.26	2,466,700	373.26
8	24-Aug-06	374.44	376.4	372.26	373.73	3,482,500	373.73
9	23-Aug-06	377.64	378.27	372.66	373.43	3,642,300	373.43
10	22-Aug-06	377.73	379.26	374.84	378.29	4,164,100	378.29
11	21-Aug-06	378.1	379	375.22	377.3	4,023,300	377.3
12	18-Aug-06	386.31	387.09	380.75	383.36	4,952,200	383.36
13	17-Aug-06	386.39	390	383.92	385.8	5,080,200	385.8
14	16-Aug-06	383.48	388.45	382.12	387.72	5,853,200	387.72
15	15-Aug-06	374.11	381.67	372.6	380.97	6,698,200	380.97
16	14-Aug-06	371.5	375.13	368.67	369.43	4,968,300	369.43
17	11-Aug-06	374.4	375.28	368	368.5	3,766,500	368.5
18	10-Aug-06	373.88	377.67	372.46	374.2	4,261,900	374.2
19	9-Aug-06	382.8	384.68	376.36	376.94	4,311,000	376.94
20	8-Aug-06	382.82	384.5	379.09	381	5,743,200	381
21	7-Aug-06	371.5	379.73	371.15	377.95	3,946,900	377.95
22	4-Aug-06	379.56	380.68	371.75	373.85	5,095,200	373.85
23	3-Aug-06	364.98	377.91	363.36	375.39	6,327,000	375.39
24	2-Aug-06	375.6	377.17	365.2	367.23	7,097,800	367.23
25	1-Aug-06	385.11	385.77	375.51	375.51	5,463,200	375.51
26	31-Jul-06	388	389.17	383.31	386.6	4,595,300	386.6
27	28-Jul-06	382	389.56	381.73	388.12	4,083,600	388.12
28	27-Jul-06	387.37	387.49	377.95	382.4	5,641,100	382.4
29	26-Jul-06	388.2	391.91	383	385.5	5,631,900	385.5
30	25-Jul-06	385.02	391.31	383.8	389.36	5,761,100	389.36
31	24-Jul-06	392.82	393.89	381.21	390.9	8,086,100	390.9
32	21-Jul-06	386.14	391.75	377.69	390.11	11,754,600	390.11
33	20-Jul-06	404.28	404.44	385.66	387.12	12,538,700	387.12

Figure 3.7: Pulling Data into MS Excel Worksheet

3.2.2 CUSTOMIZING THE WEB QUERY

After we have created a Web query, we can customize it to meet our needs through accessing Web query properties. By making a right-click on a cell in the query results and choose Edit Query, or click Edit Query on the External Data toolbar, or choose Data | Import External Data | Edit Query. When the Web page that we are querying appears, we click the Options button in the upper-right corner of the window to open the dialog box shown in **Figure 3.8**. The options here allow us to change how the query interacts with the Web page itself.

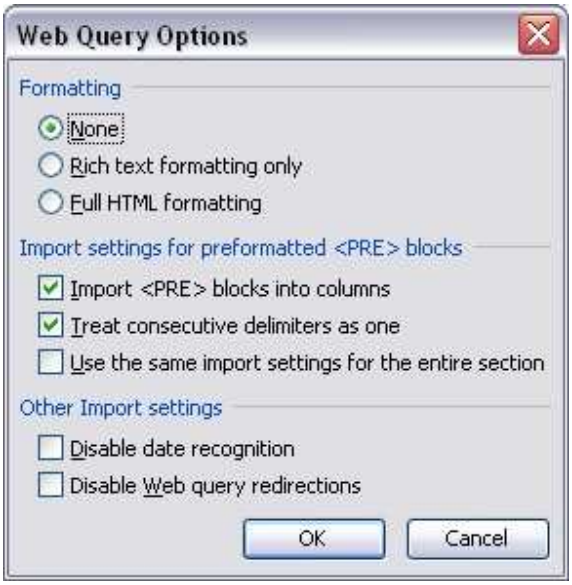


Figure 3.8: Web Query Options Dialog Box

In addition, we can edit the Data Range options to meet our needs, and we have the same choice of Data Range options that we could have with other external data queries, such as ODBC queries. Just right-click on a cell in the query results and then choose Data Range Properties, or click Edit Query on the External Data toolbar or choose Data | Import External Data | Data Range Properties, to open the dialog box shown in **Figure 3.9**. For example, we will probably want to change the Data Range's name from the default to a name we can easily access and memorize.

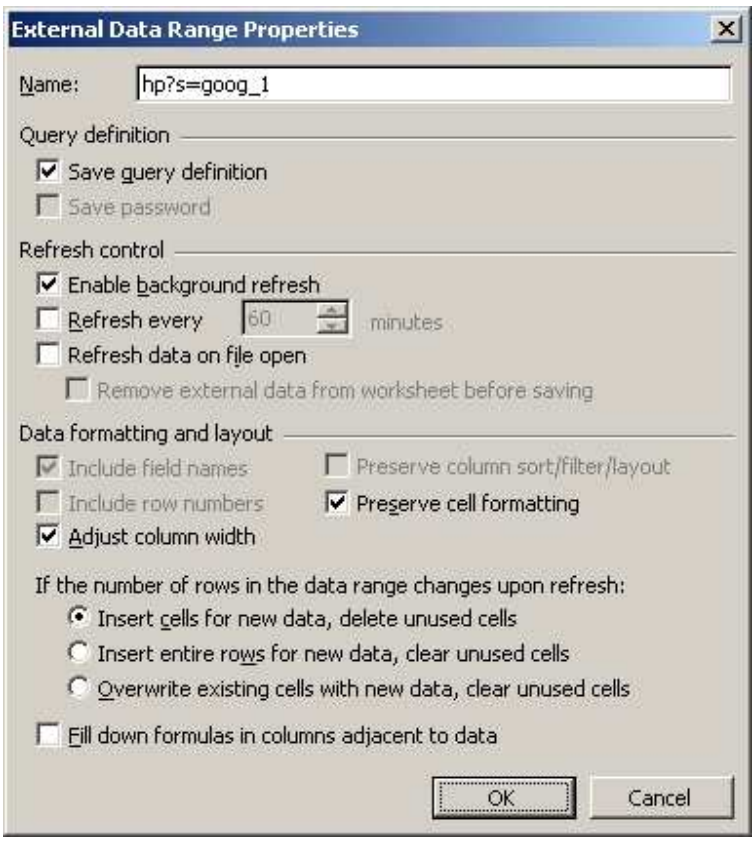


Figure 3.9: External Data Range Properties Dialog Box

3.3 MODIFYING AND REFORMATTING EXCEL TABLES

Many times, the Web page we are interested in may not contain well-formatted tables as the example we have mentioned in the previous section. Lots of times, we found that the data we need to pull to the excel sheet, is scattered in lists that is not really tables.

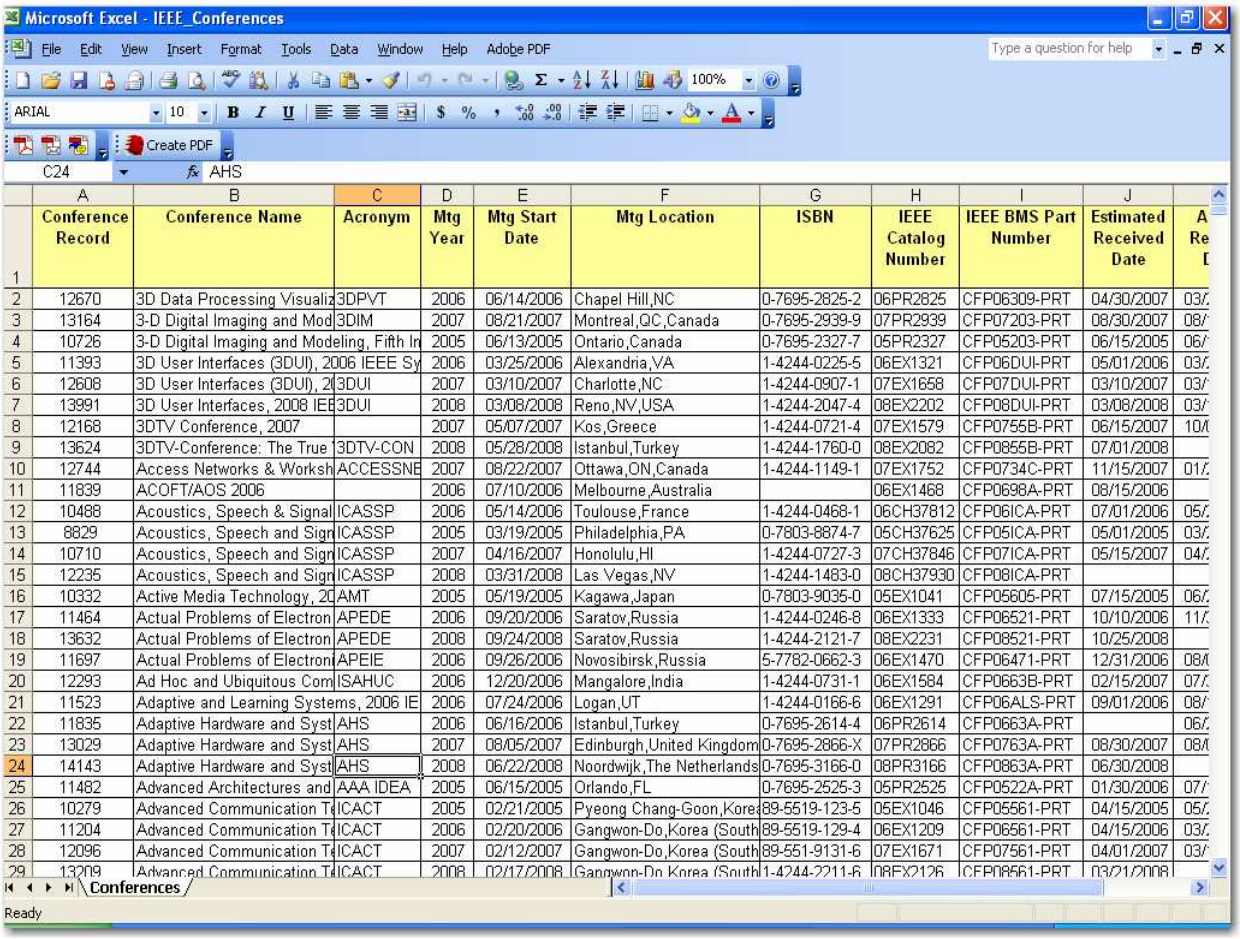
To deal with this problem, we must make some modifications and reformatting to the data in the excel sheet and put it in a well-formatted tables, and some times add headers to the tables and also a Primary Key (PK). The reason we are going to do this is to make those tables and their data more readable and organized. And most important, is to make those tables able to be converted to a relational database later on.

In this section, we are going to view the excel sheets we worked with from the Web sites we have mentioned previously in their final format, after being modified and reformatted, and that will be converted to our database later on.

We have named the tables as the following: Conferences, Journals, Magazines, Educational_Courses, Articles, Citations, Books, Affiliated_Organizations, Newsletters, Proceedings, Special_Interest_Groups and Transactions. And now we will list all the tables in the figures shown below, from **Figure 3.10** to **Figure 3.21** respectively.

Figure 3.10 shows Conferences worksheet. We have imported Conferences table from IEEE Web site, and it contains the following columns:

1. *Conference Record*: This attribute indicates a unique number for each conference, and will be the table's Primary key when it will be imported to the database.
2. *Conference Name*: This attribute contains the conference name.
3. *Acronym*: This attribute indicates an abbreviation for the conference name.
4. *Mtg Year*: This attribute indicates the meeting year.
5. *Mtg Start Date*: This attribute indicates the meeting starting date.
6. *Mtg Location*: This attribute indicates the meeting location.
7. *ISBN*: This attribute contains a 10-digit number for the conference.
8. *IEEE Catalog No*: This attribute contains the conference's IEEE catalog number.
9. *IEEE BMS Part No*: This attribute contains the conference's IEEE BMS Part number.
10. *Estimated Received Date*: This attribute contains the estimated received date.
11. *Actual Release Date*: This attribute contains the actual release date.
12. *Xplore PU No*: This attribute contains the IEEE Xplore number if posted at.
13. *Posted to Xplore*: This attribute indicates if the conference is posted to the IEEE Xplore or not yet, (Y/N).

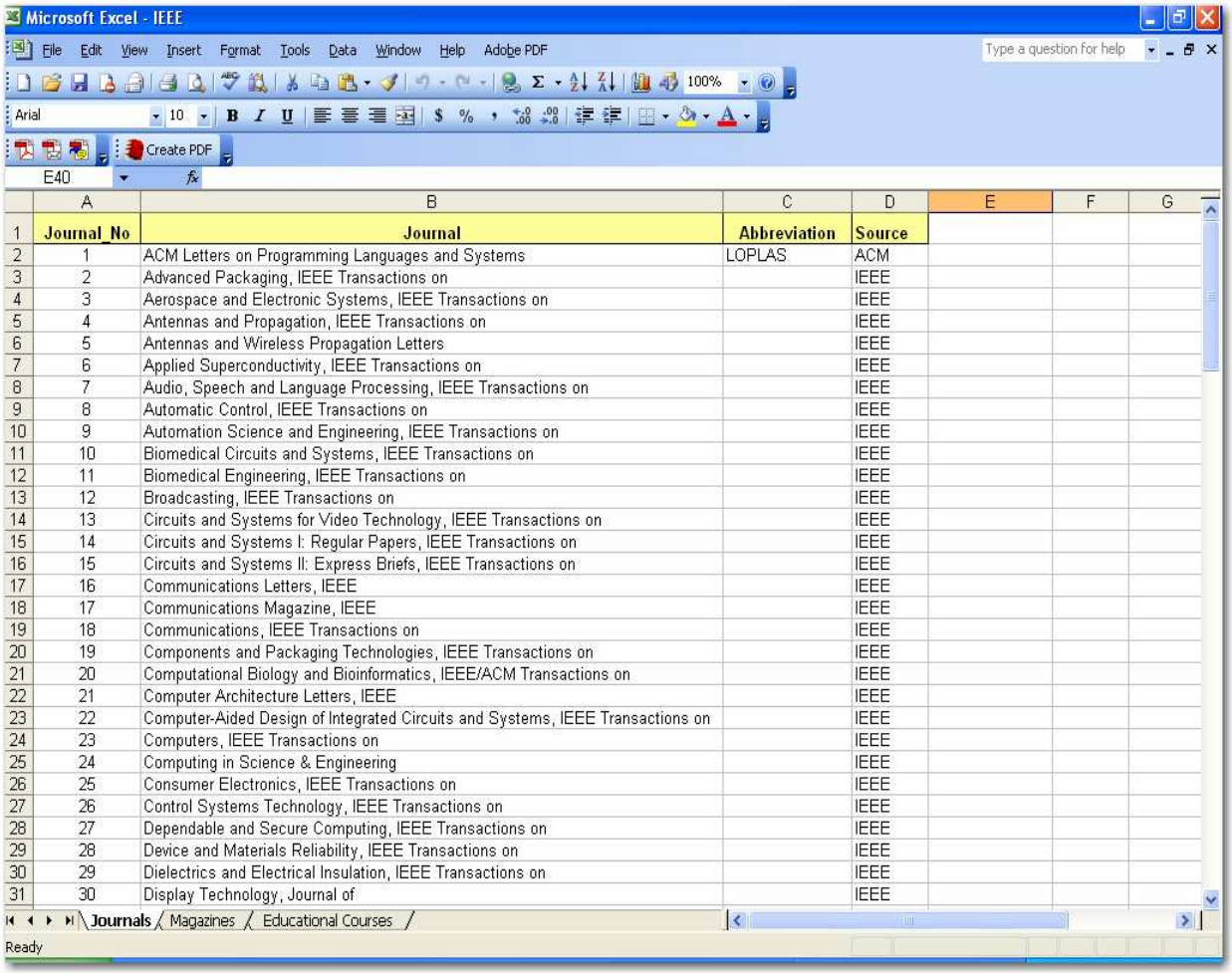


Conference Record	Conference Name	Acronym	Mtg Year	Mtg Start Date	Mtg Location	ISBN	IEEE Catalog Number	IEEE BMS Part Number	Estimated Received Date	A Re D
12670	3D Data Processing Visualiz	3DPVT	2006	06/14/2006	Chapel Hill,NC	0-7695-2825-2	06PR2825	CFP06309-PRT	04/30/2007	03/
13164	3-D Digital Imaging and Mod	3DIM	2007	08/21/2007	Montreal,QC,Canada	0-7695-2939-9	07PR2939	CFP07203-PRT	08/30/2007	08/
10726	3-D Digital Imaging and Modeling, Fifth In		2005	06/13/2005	Ontario,Canada	0-7695-2327-7	05PR2327	CFP05203-PRT	06/15/2005	06/
11393	3D User Interfaces (3DUI), 2006 IEEE Sy		2006	03/25/2006	Alexandria,VA	1-4244-0225-5	06EX1321	CFP06DUI-PRT	05/01/2006	03/
12608	3D User Interfaces (3DUI), 2(	3DUI	2007	03/10/2007	Charlotte,NC	1-4244-0907-1	07EX1658	CFP07DUI-PRT	03/10/2007	03/
13991	3D User Interfaces, 2008 IEE	3DUI	2008	03/08/2008	Reno,NV,USA	1-4244-2047-4	08EX2202	CFP08DUI-PRT	03/08/2008	03/
12168	3DTV Conference, 2007		2007	05/07/2007	Kos,Greece	1-4244-0721-4	07EX1579	CFP0755B-PRT	06/15/2007	10/
13624	3DTV-Conference: The True '	3DTV-CON	2008	05/28/2008	Istanbul,Turkey	1-4244-1760-0	08EX2082	CFP0855B-PRT	07/01/2008	
12744	Access Networks & Worksh	ACCESSNE	2007	08/22/2007	Ottawa,ON,Canada	1-4244-1149-1	07EX1752	CFP0734C-PRT	11/15/2007	01/
11839	ACOFT/AOS 2006		2006	07/10/2006	Melbourne,Australia		06EX1468	CFP0698A-PRT	08/15/2006	
10488	Acoustics, Speech & Signal	ICASSP	2006	05/14/2006	Toulouse,France	1-4244-0468-1	06CH37812	CFP06ICA-PRT	07/01/2006	05/
8829	Acoustics, Speech and Sign	ICASSP	2005	03/19/2005	Philadelphia,PA	0-7803-8874-7	05CH37625	CFP05ICA-PRT	05/01/2005	03/
10710	Acoustics, Speech and Sign	ICASSP	2007	04/16/2007	Honolulu,HI	1-4244-0727-3	07CH37846	CFP07ICA-PRT	05/15/2007	04/
12235	Acoustics, Speech and Sign	ICASSP	2008	03/31/2008	Las Vegas,NV	1-4244-1483-0	08CH37930	CFP08ICA-PRT		
10332	Active Media Technology, 2(	AMT	2005	05/19/2005	Kagawa,Japan	0-7803-9035-0	05EX1041	CFP05605-PRT	07/15/2005	06/
11464	Actual Problems of Electron	APEDE	2006	09/20/2006	Saratov,Russia	1-4244-0246-8	06EX1333	CFP06521-PRT	10/10/2006	11/
13632	Actual Problems of Electron	APEDE	2008	09/24/2008	Saratov,Russia	1-4244-2121-7	08EX2231	CFP08521-PRT	10/25/2008	
11697	Actual Problems of Electron	APEIE	2006	09/26/2006	Novosibirsk,Russia	5-7782-0662-3	06EX1470	CFP06471-PRT	12/31/2006	08/
12293	Ad Hoc and Ubiquitous Com	ISAHUC	2006	12/20/2006	Mangalore,India	1-4244-0731-1	06EX1584	CFP0663B-PRT	02/15/2007	07/
11523	Adaptive and Learning Systems, 2006 IE		2006	07/24/2006	Logan,UT	1-4244-0166-6	06EX1291	CFP06ALS-PRT	09/01/2006	08/
11835	Adaptive Hardware and Syst	AHS	2006	06/16/2006	Istanbul,Turkey	0-7695-2614-4	06PR2614	CFP0663A-PRT		06/
13029	Adaptive Hardware and Syst	AHS	2007	08/05/2007	Edinburgh,United Kingdom	0-7695-2866-X	07PR2866	CFP0763A-PRT	08/30/2007	08/
14143	Adaptive Hardware and Syst	AHS	2008	06/22/2008	Noordwijk,The Netherlands	0-7695-3166-0	08PR3166	CFP0863A-PRT	06/30/2008	
11482	Advanced Architectures and	AAA IDEA	2005	06/15/2005	Orlando,FL	0-7695-2525-3	05PR2525	CFP0522A-PRT	01/30/2006	07/
10279	Advanced Communication Te	ICACT	2005	02/21/2005	Pyeong Chang-Goon,Korea	89-5519-123-5	05EX1046	CFP05561-PRT	04/15/2005	05/
11204	Advanced Communication Te	ICACT	2006	02/20/2006	Gangwon-Do,Korea (South	89-5519-129-4	06EX1209	CFP06561-PRT	04/15/2006	03/
12096	Advanced Communication Te	ICACT	2007	02/12/2007	Gangwon-Do,Korea (South	89-551-9131-6	07EX1671	CFP07561-PRT	04/01/2007	03/
13209	Advanced Communication Te	ICACT	2008	02/17/2008	Gangwon-Do,Korea (South	1-4244-2211-6	08EX2126	CFP08561-PRT	03/21/2008	

Figure 3.10: Conferences Excel Sheet

Figure 3.11 shows Journals worksheet. We have imported the Journals table from IEEE Web site and also the ACM Web site and merge their data together, and it contains the following columns:

1. *Journal_No*: This attribute indicates a unique number for each Journal, and will be the table's Primary key when it will be imported to the database.
2. *Journal*: This attribute contains the Journal name.
3. *Abbreviation*: This attribute indicates an abbreviation for the Journal name.
4. *Source*: This attribute indicates the source of the Journal, (IEEE/ACM).

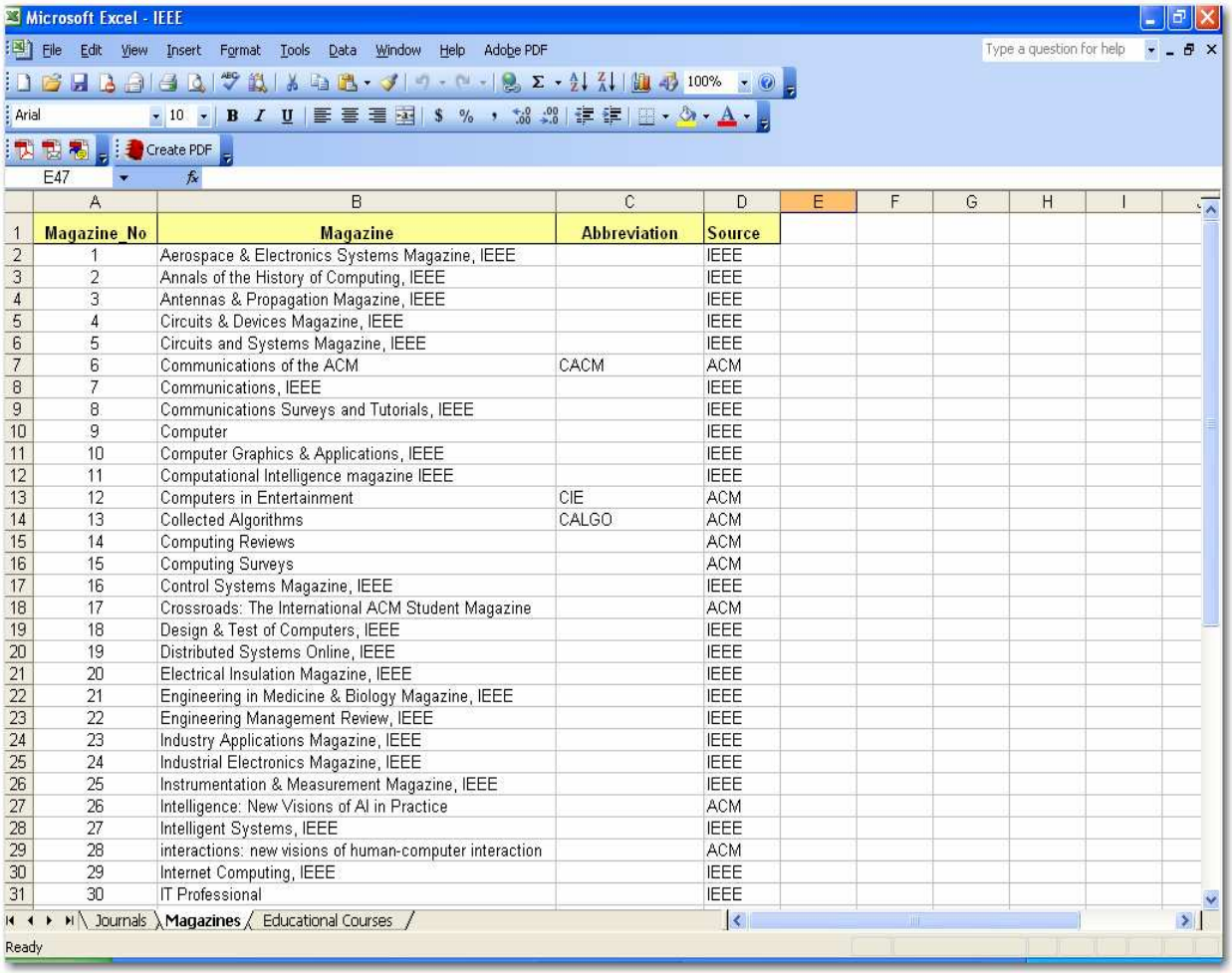


Journal_No	Journal	Abbreviation	Source
1	ACM Letters on Programming Languages and Systems	LOPLAS	ACM
2	Advanced Packaging, IEEE Transactions on		IEEE
3	Aerospace and Electronic Systems, IEEE Transactions on		IEEE
4	Antennas and Propagation, IEEE Transactions on		IEEE
5	Antennas and Wireless Propagation Letters		IEEE
6	Applied Superconductivity, IEEE Transactions on		IEEE
7	Audio, Speech and Language Processing, IEEE Transactions on		IEEE
8	Automatic Control, IEEE Transactions on		IEEE
9	Automation Science and Engineering, IEEE Transactions on		IEEE
10	Biomedical Circuits and Systems, IEEE Transactions on		IEEE
11	Biomedical Engineering, IEEE Transactions on		IEEE
12	Broadcasting, IEEE Transactions on		IEEE
13	Circuits and Systems for Video Technology, IEEE Transactions on		IEEE
14	Circuits and Systems I: Regular Papers, IEEE Transactions on		IEEE
15	Circuits and Systems II: Express Briefs, IEEE Transactions on		IEEE
16	Communications Letters, IEEE		IEEE
17	Communications Magazine, IEEE		IEEE
18	Communications, IEEE Transactions on		IEEE
19	Components and Packaging Technologies, IEEE Transactions on		IEEE
20	Computational Biology and Bioinformatics, IEEE/ACM Transactions on		IEEE
21	Computer Architecture Letters, IEEE		IEEE
22	Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on		IEEE
23	Computers, IEEE Transactions on		IEEE
24	Computing in Science & Engineering		IEEE
25	Consumer Electronics, IEEE Transactions on		IEEE
26	Control Systems Technology, IEEE Transactions on		IEEE
27	Dependable and Secure Computing, IEEE Transactions on		IEEE
28	Device and Materials Reliability, IEEE Transactions on		IEEE
29	Dielectrics and Electrical Insulation, IEEE Transactions on		IEEE
30	Display Technology, Journal of		IEEE

Figure 3.11: Journals Excel Sheet

Figure 3.12 shows Magazines worksheet. We have imported the Magazines table from IEEE Web site and also the ACM Web site and merge their data together, and it contains the following columns:

1. *Magazine_No*: This attribute indicates a unique number for each Magazine, and will be the table's Primary key when it will be imported to the database.
2. *Magazine*: This attribute contains the Magazine name.
3. *Abbreviation*: This attribute indicates an abbreviation for the Magazine name.
4. *Source*: This attribute indicates the source of the Magazine, (IEEE/ACM).

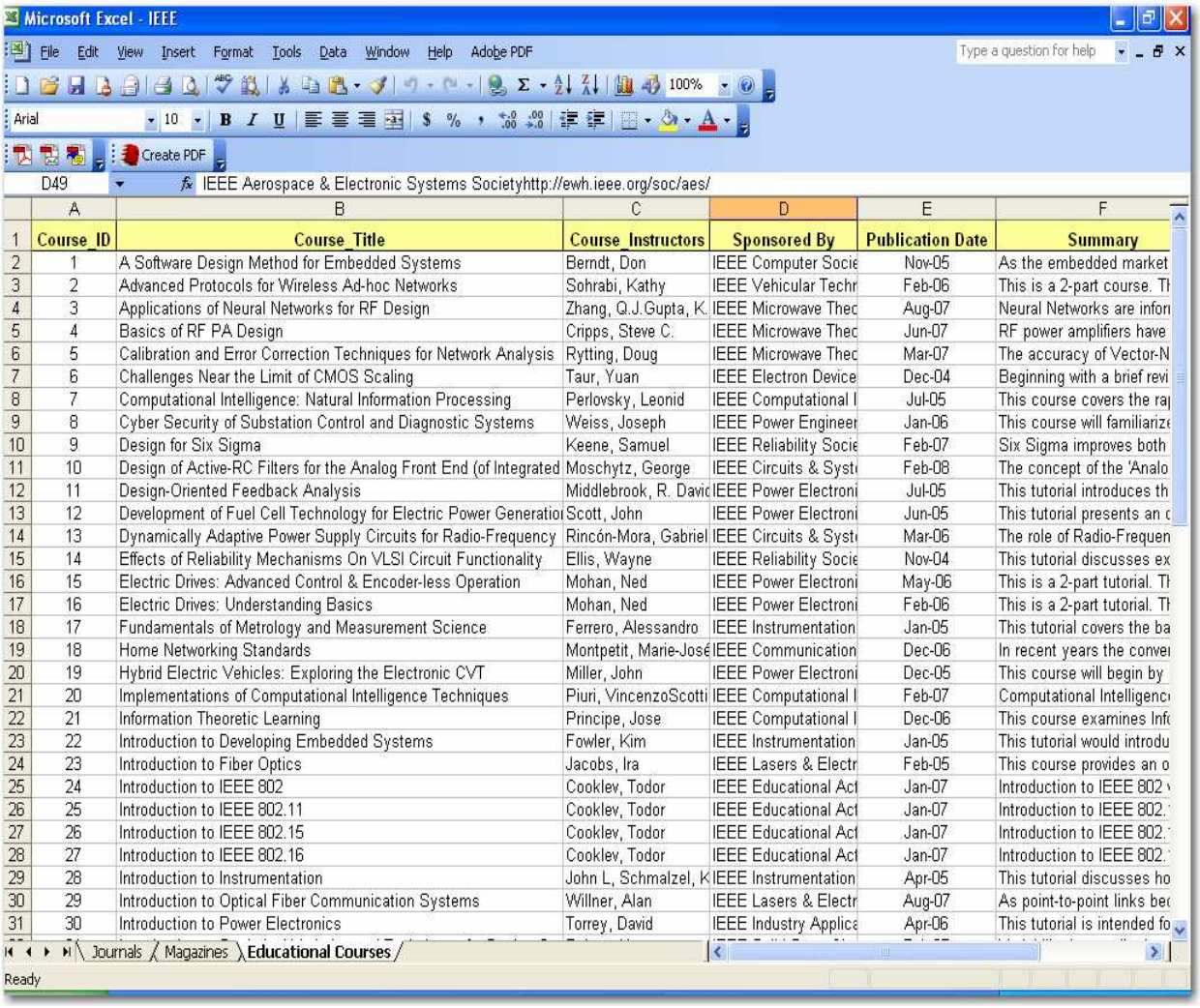


	A	B	C	D
1	Magazine_No	Magazine	Abbreviation	Source
2	1	Aerospace & Electronics Systems Magazine, IEEE		IEEE
3	2	Annals of the History of Computing, IEEE		IEEE
4	3	Antennas & Propagation Magazine, IEEE		IEEE
5	4	Circuits & Devices Magazine, IEEE		IEEE
6	5	Circuits and Systems Magazine, IEEE		IEEE
7	6	Communications of the ACM	CACM	ACM
8	7	Communications, IEEE		IEEE
9	8	Communications Surveys and Tutorials, IEEE		IEEE
10	9	Computer		IEEE
11	10	Computer Graphics & Applications, IEEE		IEEE
12	11	Computational Intelligence magazine IEEE		IEEE
13	12	Computers in Entertainment	CIE	ACM
14	13	Collected Algorithms	CALGO	ACM
15	14	Computing Reviews		ACM
16	15	Computing Surveys		ACM
17	16	Control Systems Magazine, IEEE		IEEE
18	17	Crossroads: The International ACM Student Magazine		ACM
19	18	Design & Test of Computers, IEEE		IEEE
20	19	Distributed Systems Online, IEEE		IEEE
21	20	Electrical Insulation Magazine, IEEE		IEEE
22	21	Engineering in Medicine & Biology Magazine, IEEE		IEEE
23	22	Engineering Management Review, IEEE		IEEE
24	23	Industry Applications Magazine, IEEE		IEEE
25	24	Industrial Electronics Magazine, IEEE		IEEE
26	25	Instrumentation & Measurement Magazine, IEEE		IEEE
27	26	Intelligence: New Visions of AI in Practice		ACM
28	27	Intelligent Systems, IEEE		IEEE
29	28	interactions: new visions of human-computer interaction		ACM
30	29	Internet Computing, IEEE		IEEE
31	30	IT Professional		IEEE

Figure 3.12: Magazines Excel Sheet

Figure 3.13 shows Educational Courses worksheet. We have imported the Educational Courses table from IEEE Web site, and it contains the following columns:

1. *Course_ID*: This attribute indicates a unique number for each Course, and will be the table's Primary key when it will be imported to the database.
2. *Course_Title*: This attribute contains the Course title.
3. *Course_Instructors*: This attribute contains the Course Instructors' Names.
4. *Sponsored By*: This attribute shows by whom the Course is sponsored.
5. *Publication Date*: This attribute contains the Course publication date.
6. *Summary*: This attribute contains a summary for the Course.



Microsoft Excel - IEEE

File Edit View Insert Format Tools Data Window Help Adobe PDF

D49: IEEE Aerospace & Electronic Systems Societyhttp://ewh.ieee.org/soc/aes/

	A	B	C	D	E	F
1	Course_ID	Course_Title	Course_Instructors	Sponsored By	Publication Date	Summary
2	1	A Software Design Method for Embedded Systems	Berndt, Don	IEEE Computer Socie	Nov-05	As the embedded market
3	2	Advanced Protocols for Wireless Ad-hoc Networks	Sohrabi, Kathy	IEEE Vehicular Techr	Feb-06	This is a 2-part course. Th
4	3	Applications of Neural Networks for RF Design	Zhang, Q.J.Gupta, K.	IEEE Microwave Thec	Aug-07	Neural Networks are infor
5	4	Basics of RF PA Design	Cripps, Steve C.	IEEE Microwave Thec	Jun-07	RF power amplifiers have
6	5	Calibration and Error Correction Techniques for Network Analysis	Rytting, Doug	IEEE Microwave Thec	Mar-07	The accuracy of Vector-N
7	6	Challenges Near the Limit of CMOS Scaling	Taur, Yuan	IEEE Electron Device	Dec-04	Beginning with a brief revi
8	7	Computational Intelligence: Natural Information Processing	Perlovsky, Leonid	IEEE Computational I	Jul-05	This course covers the ra
9	8	Cyber Security of Substation Control and Diagnostic Systems	Weiss, Joseph	IEEE Power Engineer	Jan-06	This course will familiariz
10	9	Design for Six Sigma	Keene, Samuel	IEEE Reliability Socie	Feb-07	Six Sigma improves both
11	10	Design of Active-RC Filters for the Analog Front End (of Integrated	Moschytz, George	IEEE Circuits & Syst	Feb-08	The concept of the 'Analo
12	11	Design-Oriented Feedback Analysis	Middlebrook, R. Davic	IEEE Power Electroni	Jul-05	This tutorial introduces th
13	12	Development of Fuel Cell Technology for Electric Power Generatio	Scott, John	IEEE Power Electroni	Jun-05	This tutorial presents an c
14	13	Dynamically Adaptive Power Supply Circuits for Radio-Frequency	Rincón-Mora, Gabriel	IEEE Circuits & Syst	Mar-06	The role of Radio-Frequen
15	14	Effects of Reliability Mechanisms On VLSI Circuit Functionality	Ellis, Wayne	IEEE Reliability Socie	Nov-04	This tutorial discusses ex
16	15	Electric Drives: Advanced Control & Encoder-less Operation	Mohan, Ned	IEEE Power Electroni	May-06	This is a 2-part tutorial. Th
17	16	Electric Drives: Understanding Basics	Mohan, Ned	IEEE Power Electroni	Feb-06	This is a 2-part tutorial. Th
18	17	Fundamentals of Metrology and Measurement Science	Ferrero, Alessandro	IEEE Instrumentation	Jan-05	This tutorial covers the ba
19	18	Home Networking Standards	Montpetit, Marie-José	IEEE Communication	Dec-06	In recent years the conve
20	19	Hybrid Electric Vehicles: Exploring the Electronic CVT	Miller, John	IEEE Power Electroni	Dec-05	This course will begin by
21	20	Implementations of Computational Intelligence Techniques	Piuri, VincenzoScotti	IEEE Computational I	Feb-07	Computational Intelligenc
22	21	Information Theoretic Learning	Principe, Jose	IEEE Computational I	Dec-06	This course examines Inf
23	22	Introduction to Developing Embedded Systems	Fowler, Kim	IEEE Instrumentation	Jan-05	This tutorial would introdu
24	23	Introduction to Fiber Optics	Jacobs, Ira	IEEE Lasers & Electr	Feb-05	This course provides an o
25	24	Introduction to IEEE 802	Cooklev, Todor	IEEE Educational Act	Jan-07	Introduction to IEEE 802
26	25	Introduction to IEEE 802.11	Cooklev, Todor	IEEE Educational Act	Jan-07	Introduction to IEEE 802.
27	26	Introduction to IEEE 802.15	Cooklev, Todor	IEEE Educational Act	Jan-07	Introduction to IEEE 802.
28	27	Introduction to IEEE 802.16	Cooklev, Todor	IEEE Educational Act	Jan-07	Introduction to IEEE 802.
29	28	Introduction to Instrumentation	John L, Schmalzel, K	IEEE Instrumentation	Apr-05	This tutorial discusses ho
30	29	Introduction to Optical Fiber Communication Systems	Willner, Alan	IEEE Lasers & Electr	Aug-07	As point-to-point links be
31	30	Introduction to Power Electronics	Torrey, David	IEEE Industry Applica	Apr-06	This tutorial is intended fo

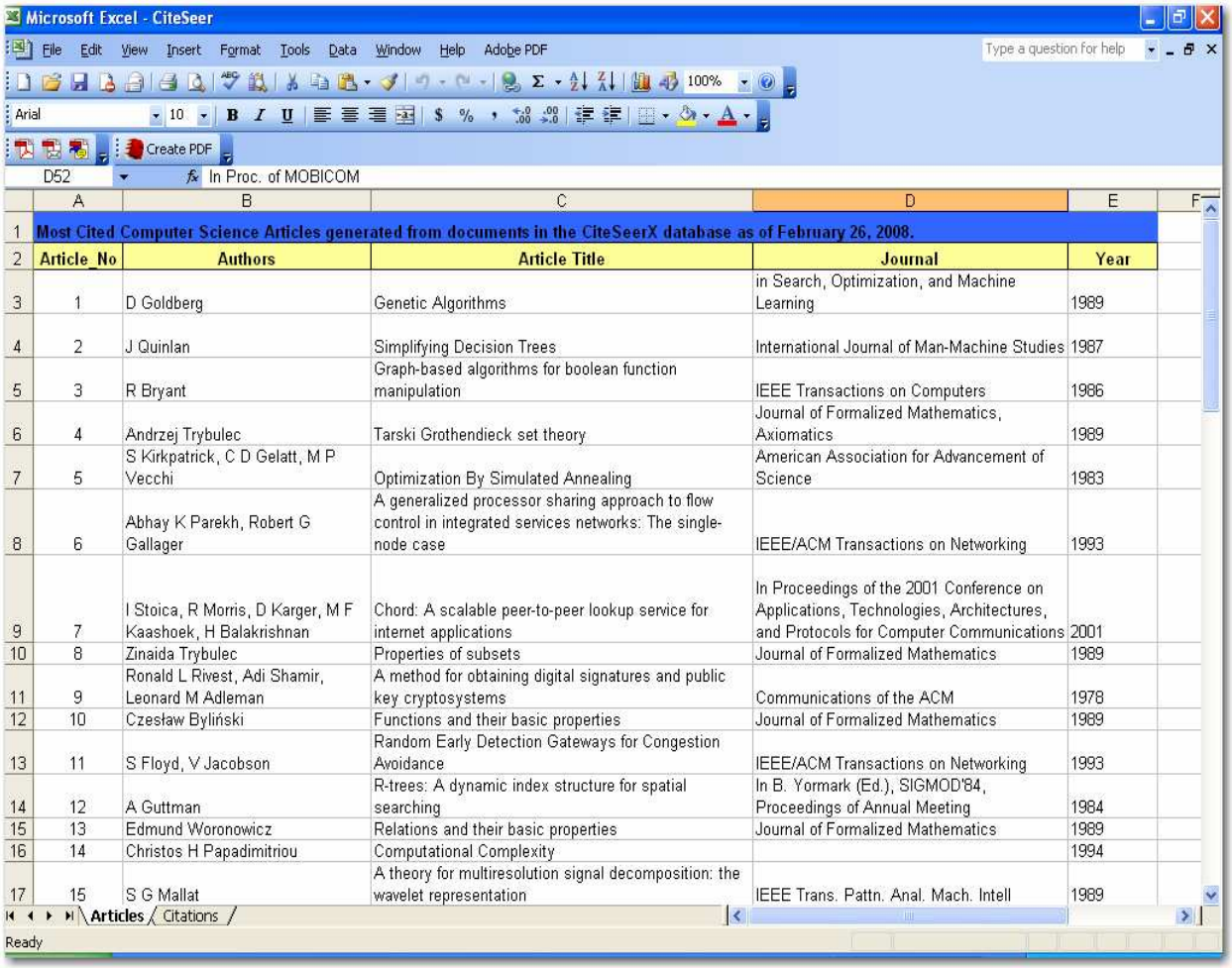
Journals / Magazines / Educational Courses

Ready

Figure 3.13: Educational_Courses Excel Sheet

Figure 3.14 shows Articles worksheet. We have imported the Articles table from CiteSeer Web site, and it contains the following columns:

1. *Article_No*: This attribute indicates a unique number for each Article, and will be the table's Primary key when it will be imported to the database.
2. *Authors*: This attribute contains the Article authors' names.
3. *Article Title*: This attribute contains the Article title.
4. *Journal*: This attribute contains the name of the journal that published the Article.
5. *Year*: This attribute contains the Article publication year.



Microsoft Excel - CiteSeer

File Edit View Insert Format Tools Data Window Help Adobe PDF — Type a question for help

D52 — In Proc. of MOBICOM

	A	B	C	D	E
1	Most Cited Computer Science Articles generated from documents in the CiteSeerX database as of February 26, 2008.				
2	Article_No	Authors	Article Title	Journal	Year
3	1	D Goldberg	Genetic Algorithms	in Search, Optimization, and Machine Learning	1989
4	2	J Quinlan	Simplifying Decision Trees	International Journal of Man-Machine Studies	1987
5	3	R Bryant	Graph-based algorithms for boolean function manipulation	IEEE Transactions on Computers	1986
6	4	Andrzej Trybulec	Tarski Grothendieck set theory	Journal of Formalized Mathematics, Axiomatics	1989
7	5	S Kirkpatrick, C D Gelatt, M P Vecchi	Optimization By Simulated Annealing	American Association for Advancement of Science	1983
8	6	Abhay K Parekh, Robert G Gallager	A generalized processor sharing approach to flow control in integrated services networks: The single-node case	IEEE/ACM Transactions on Networking	1993
9	7	I Stoica, R Morris, D Karger, M F Kaashoek, H Balakrishnan	Chord: A scalable peer-to-peer lookup service for internet applications	In Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications	2001
10	8	Zinaida Trybulec	Properties of subsets	Journal of Formalized Mathematics	1989
11	9	Ronald L Rivest, Adi Shamir, Leonard M Adleman	A method for obtaining digital signatures and public key cryptosystems	Communications of the ACM	1978
12	10	Czesław Byliński	Functions and their basic properties	Journal of Formalized Mathematics	1989
13	11	S Floyd, V Jacobson	Random Early Detection Gateways for Congestion Avoidance	IEEE/ACM Transactions on Networking	1993
14	12	A Guttman	R-trees: A dynamic index structure for spatial searching	In B. Yormark (Ed.), SIGMOD'84, Proceedings of Annual Meeting	1984
15	13	Edmund Woronowicz	Relations and their basic properties	Journal of Formalized Mathematics	1989
16	14	Christos H Papadimitriou	Computational Complexity		1994
17	15	S G Mallat	A theory for multiresolution signal decomposition: the wavelet representation	IEEE Trans. Pattn. Anal. Mach. Intell	1989

Articles / Citations

Ready

Figure 3.14: Articles Excel Sheet

Figure 3.15 shows Citations worksheet. We have imported the Citations table from CiteSeer Web site, and it contains the following columns:

1. *Citation_No*: This attribute indicates a unique number for each Citation, and will be the table's Primary key when it will be imported to the database.
2. *Authors*: This attribute contains the Citation authors' names.
3. *Citation Title*: This attribute contains the Citation title.
4. *Journal*: This attribute contains the name of the journal that published the Citation.
5. *Year*: This attribute contains the Citation publication year.

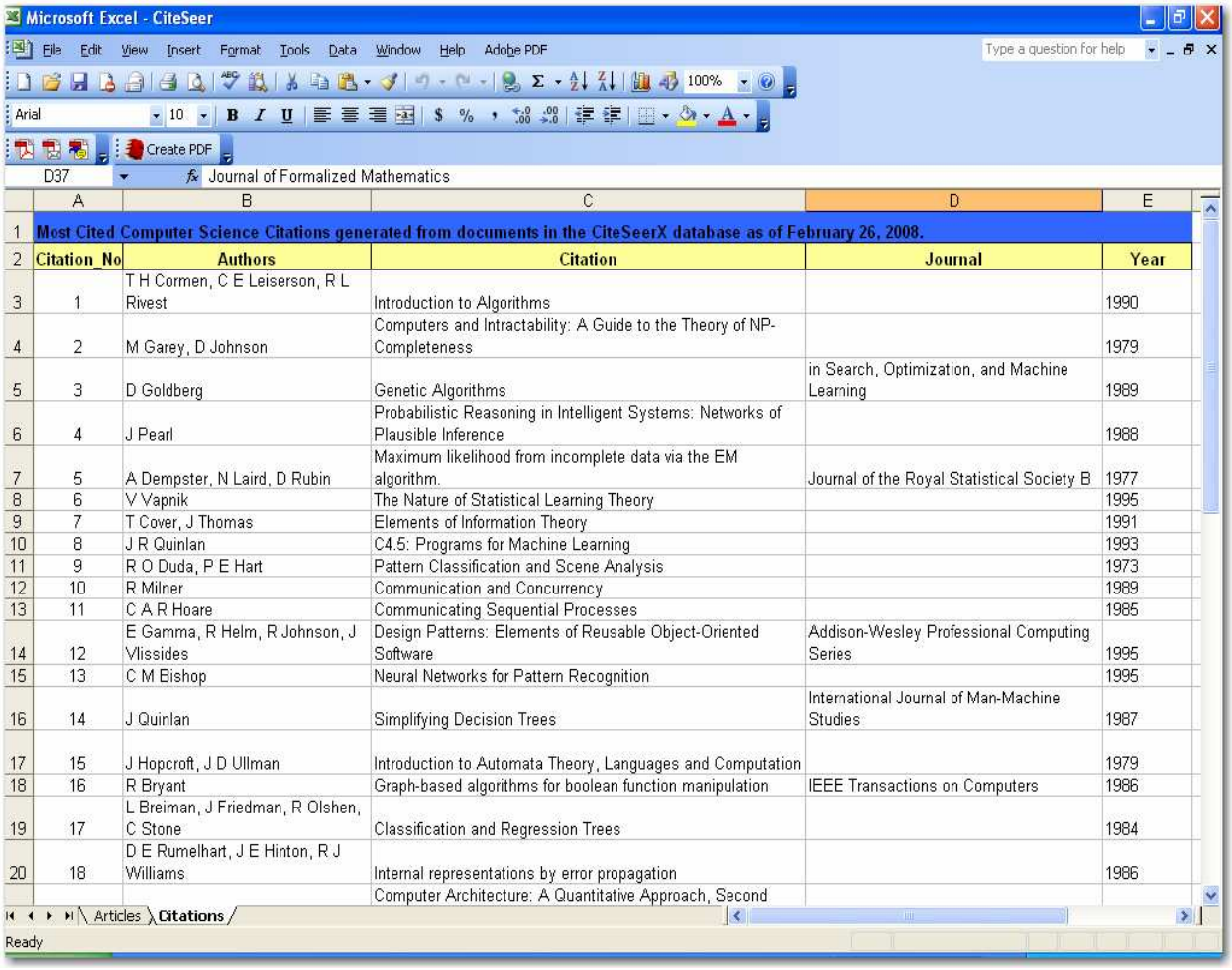
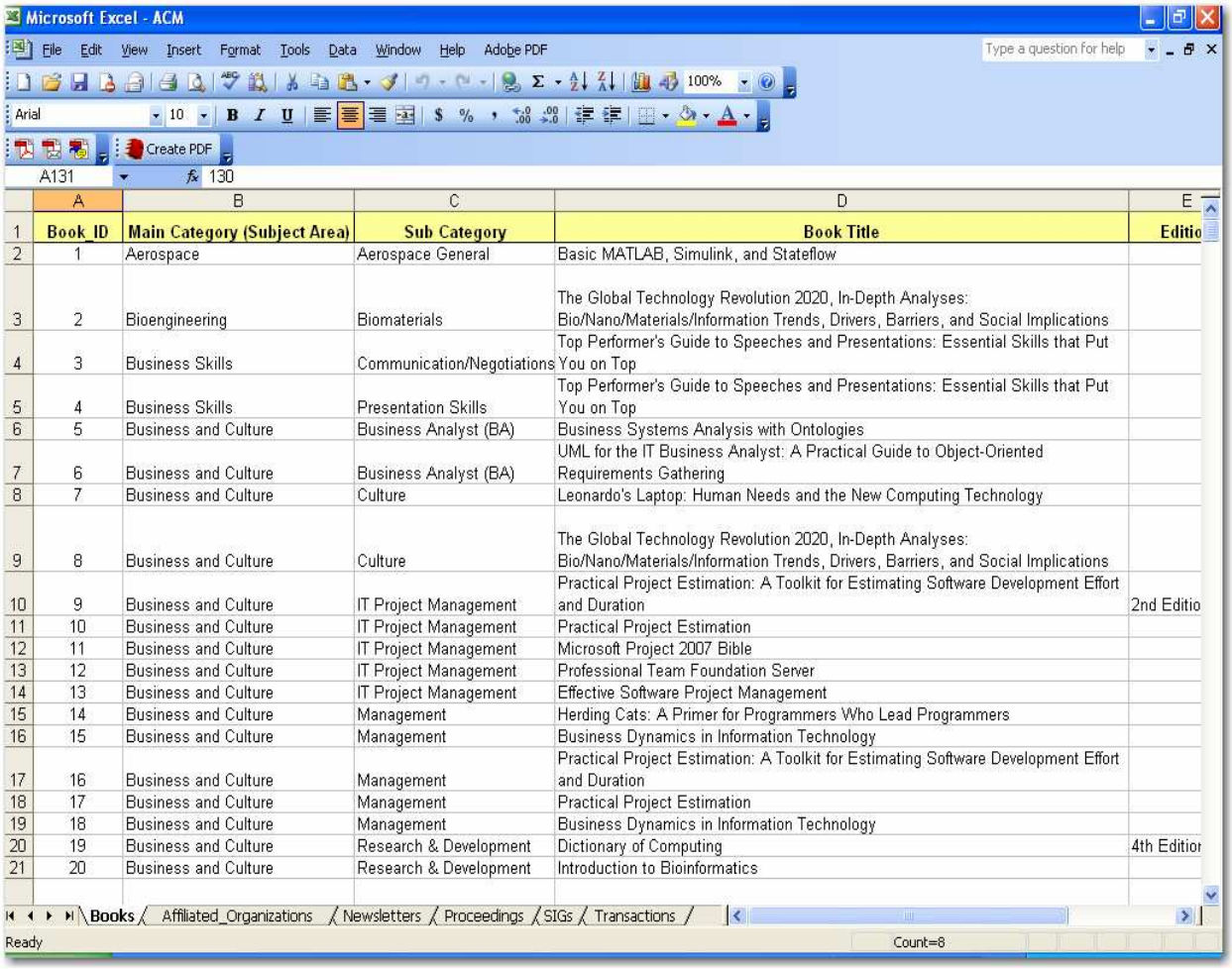


Figure 3.15: Citations Excel Sheet

Figure 3.16 shows Books worksheet. We have imported the Books table from ACM Web site, and it contains the following columns:

1. *Book_ID*: This attribute indicates a unique number for each Book, and will be the table's Primary key when it will be imported to the database.
2. *Main Category*: This attribute contains a main subject area the Book related to.
3. *Sub Category*: This attribute contains a sub subject area the Book related to.
4. *Book Title*: This attribute contains the name of the Book
5. *Edition*: This attribute contains the Edition of the Book.



Microsoft Excel - ACM

File Edit View Insert Format Tools Data Window Help Adobe PDF

Type a question for help

Arial 10 100%

Create PDF

A131 130

	A	B	C	D	E
1	Book_ID	Main Category (Subject Area)	Sub Category	Book Title	Editio
2	1	Aerospace	Aerospace General	Basic MATLAB, Simulink, and Stateflow	
3	2	Bioengineering	Biomaterials	The Global Technology Revolution 2020, In-Depth Analyses: Bio/Nano/Materials/Information Trends, Drivers, Barriers, and Social Implications	
4	3	Business Skills	Communication/Negotiations	Top Performer's Guide to Speeches and Presentations: Essential Skills that Put You on Top	
5	4	Business Skills	Presentation Skills	Top Performer's Guide to Speeches and Presentations: Essential Skills that Put You on Top	
6	5	Business and Culture	Business Analyst (BA)	Business Systems Analysis with Ontologies	
7	6	Business and Culture	Business Analyst (BA)	UML for the IT Business Analyst: A Practical Guide to Object-Oriented Requirements Gathering	
8	7	Business and Culture	Culture	Leonardo's Laptop: Human Needs and the New Computing Technology	
9	8	Business and Culture	Culture	The Global Technology Revolution 2020, In-Depth Analyses: Bio/Nano/Materials/Information Trends, Drivers, Barriers, and Social Implications	
10	9	Business and Culture	IT Project Management	Practical Project Estimation: A Toolkit for Estimating Software Development Effort and Duration	2nd Editio
11	10	Business and Culture	IT Project Management	Practical Project Estimation	
12	11	Business and Culture	IT Project Management	Microsoft Project 2007 Bible	
13	12	Business and Culture	IT Project Management	Professional Team Foundation Server	
14	13	Business and Culture	IT Project Management	Effective Software Project Management	
15	14	Business and Culture	Management	Herding Cats: A Primer for Programmers Who Lead Programmers	
16	15	Business and Culture	Management	Business Dynamics in Information Technology	
17	16	Business and Culture	Management	Practical Project Estimation: A Toolkit for Estimating Software Development Effort and Duration	
18	17	Business and Culture	Management	Practical Project Estimation	
19	18	Business and Culture	Management	Business Dynamics in Information Technology	
20	19	Business and Culture	Research & Development	Dictionary of Computing	4th Editio
21	20	Business and Culture	Research & Development	Introduction to Bioinformatics	

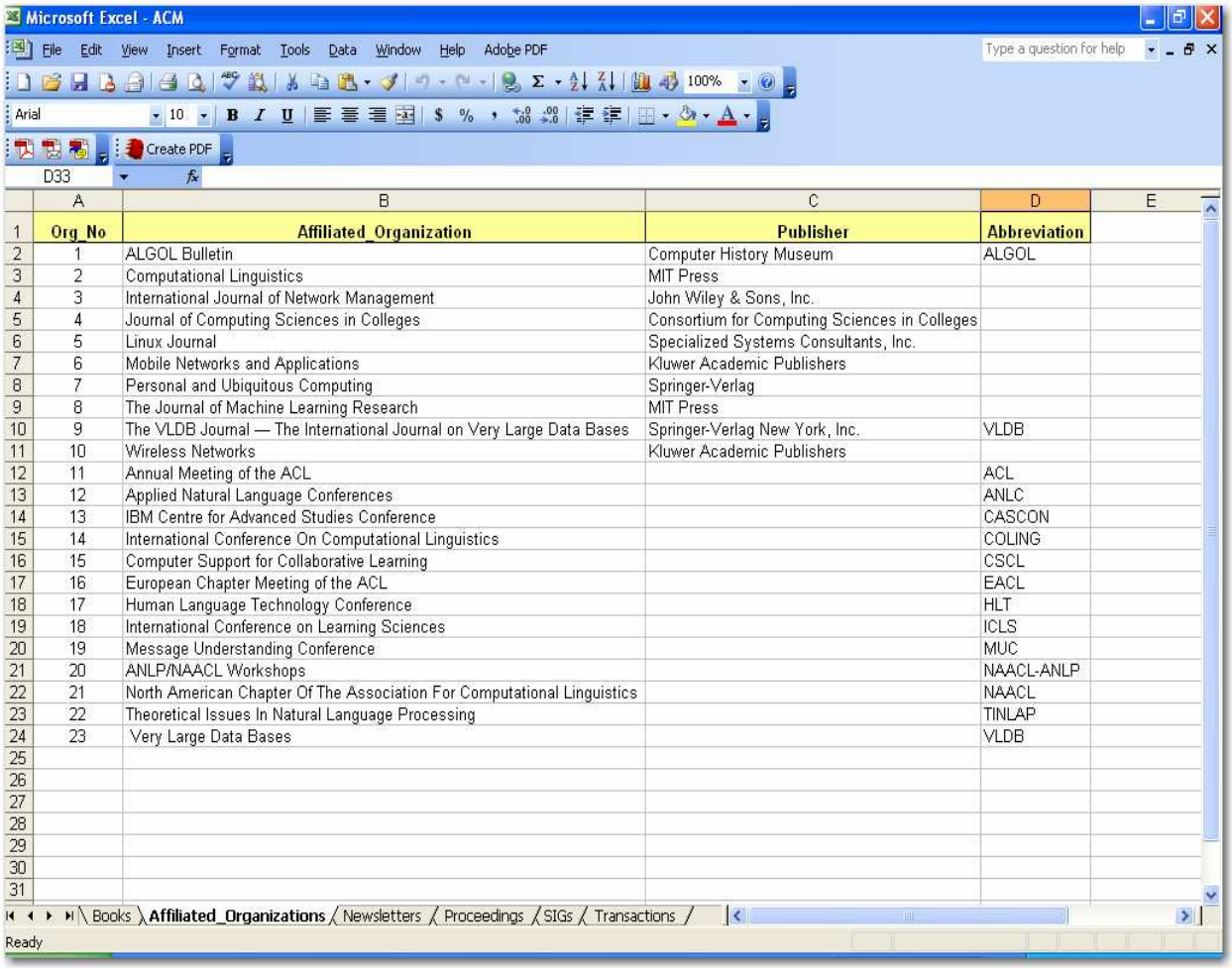
Books Affiliated_Organizations Newsletters Proceedings SIGs Transactions

Ready Count=8

Figure 3.16: Books Excel Sheet

Figure 3.17 shows Affiliated Organizations worksheet. We have imported the Affiliated Organizations table from ACM Web site, and it contains the following columns:

1. *Org_No*: This attribute indicates a unique number for each Affiliated Organization, and will be the table's Primary key when it will be imported to the database.
2. Affiliated_Organization: This attribute contains the Affiliated Organization name.
3. *Publisher*: This attribute contains the Publisher name.
4. *Abbreviation*: This attribute indicates an abbreviation for the Organization name.

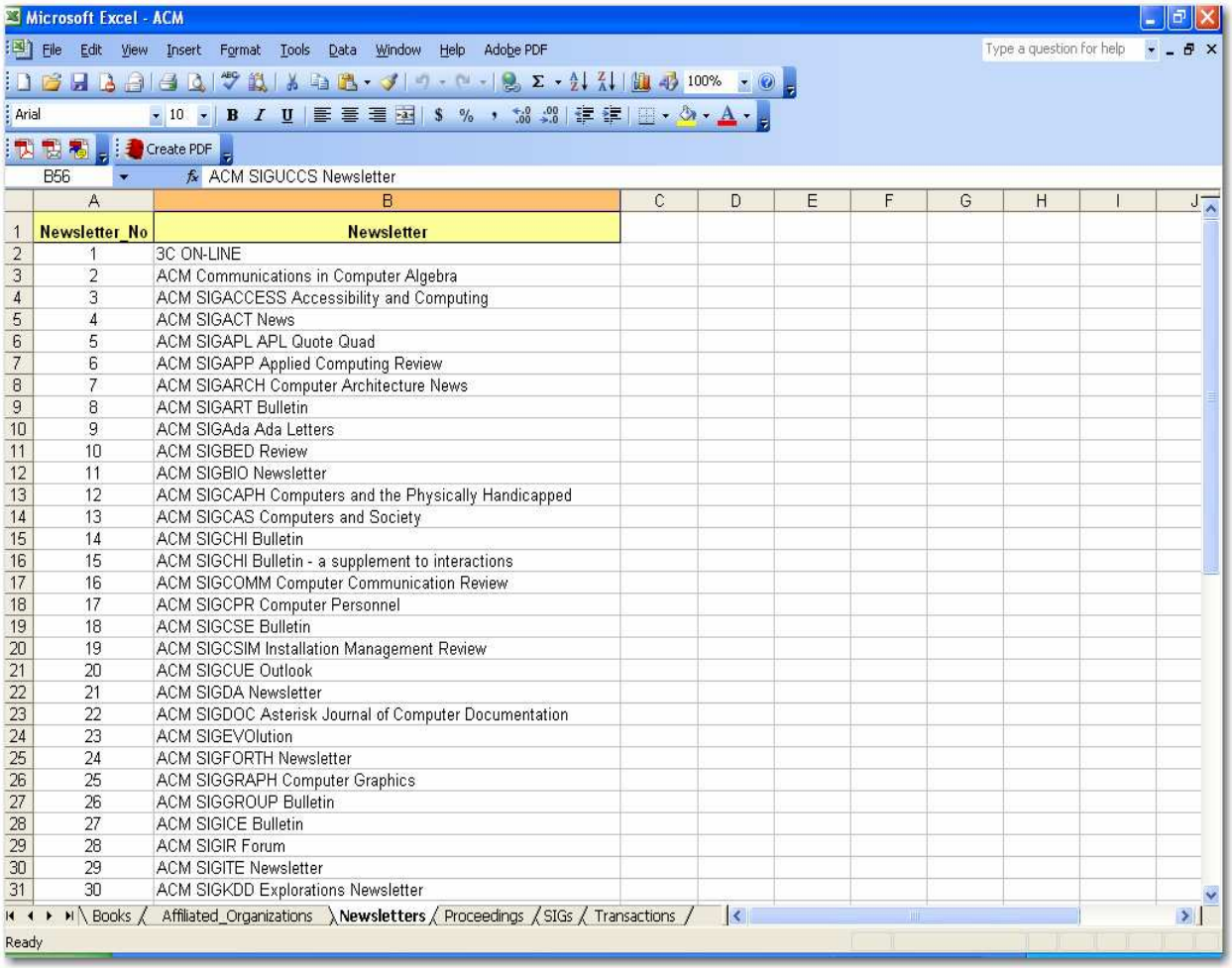


	A	B	C	D	E
1	Org_No	Affiliated_Organization	Publisher	Abbreviation	
2	1	ALGOL Bulletin	Computer History Museum	ALGOL	
3	2	Computational Linguistics	MIT Press		
4	3	International Journal of Network Management	John Wiley & Sons, Inc.		
5	4	Journal of Computing Sciences in Colleges	Consortium for Computing Sciences in Colleges		
6	5	Linux Journal	Specialized Systems Consultants, Inc.		
7	6	Mobile Networks and Applications	Kluwer Academic Publishers		
8	7	Personal and Ubiquitous Computing	Springer-Verlag		
9	8	The Journal of Machine Learning Research	MIT Press		
10	9	The VLDB Journal — The International Journal on Very Large Data Bases	Springer-Verlag New York, Inc.	VLDB	
11	10	Wireless Networks	Kluwer Academic Publishers		
12	11	Annual Meeting of the ACL		ACL	
13	12	Applied Natural Language Conferences		ANLC	
14	13	IBM Centre for Advanced Studies Conference		CASCON	
15	14	International Conference On Computational Linguistics		COLING	
16	15	Computer Support for Collaborative Learning		CSCL	
17	16	European Chapter Meeting of the ACL		EACL	
18	17	Human Language Technology Conference		HLT	
19	18	International Conference on Learning Sciences		ICLS	
20	19	Message Understanding Conference		MUC	
21	20	ANLP/NAACL Workshops		NAACL-ANLP	
22	21	North American Chapter Of The Association For Computational Linguistics		NAACL	
23	22	Theoretical Issues In Natural Language Processing		TINLAP	
24	23	Very Large Data Bases		VLDB	
25					
26					
27					
28					
29					
30					
31					

Figure 3.17: Affiliated_Organizations Excel Sheet

Figure 3.18 shows Newsletters worksheet. We have imported the Newsletters table from ACM Web site, and it contains the following columns:

1. *Newsletter_No*: This attribute indicates a unique number for each Newsletter, and will be the table's Primary key when it will be imported to the database.
2. *Newsletter*: This attribute contains the Newsletter name.

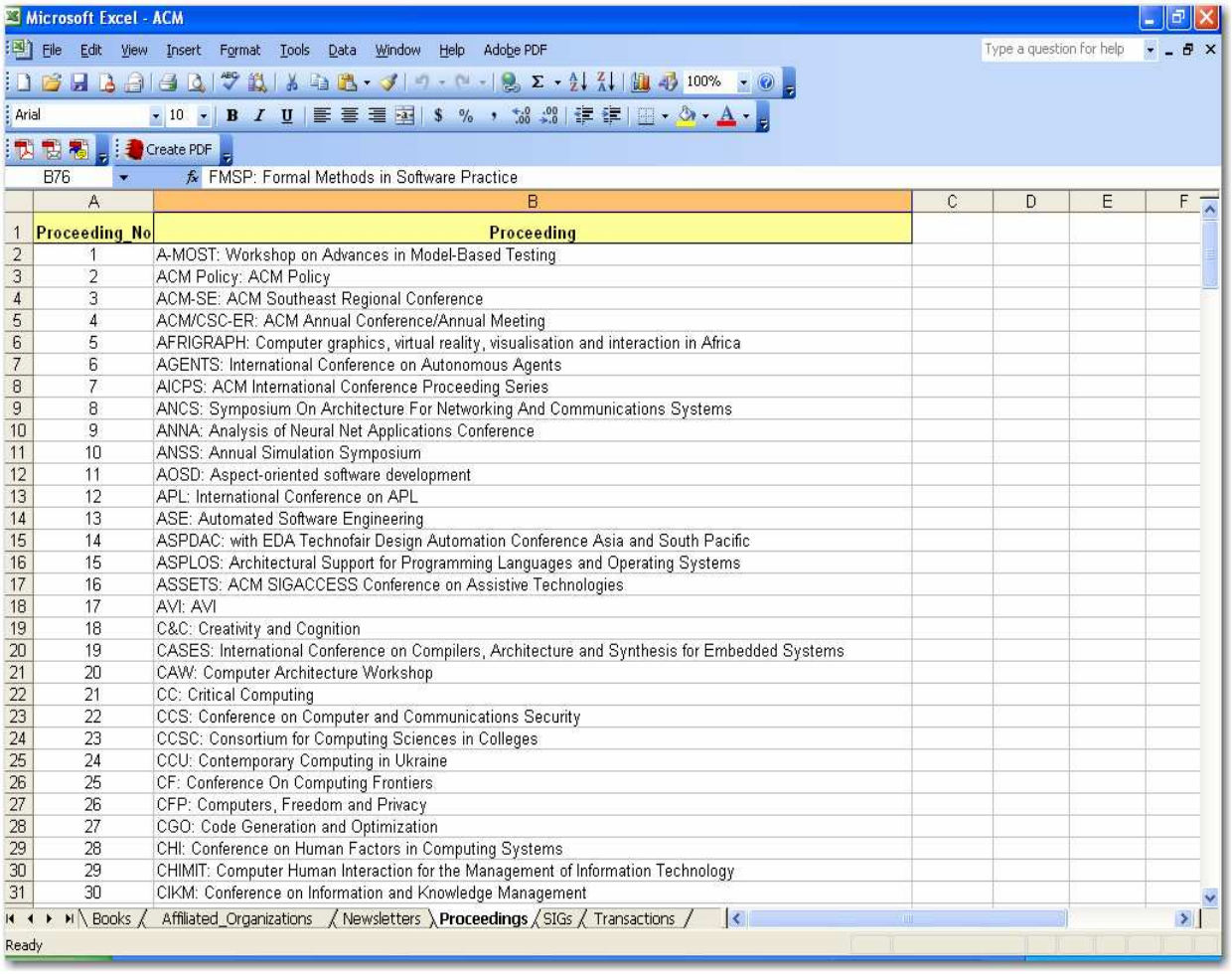


Newsletter_No	Newsletter
1	3C ON-LINE
2	ACM Communications in Computer Algebra
3	ACM SIGACCESS Accessibility and Computing
4	ACM SIGACT News
5	ACM SIGAPL APL Quote Quad
6	ACM SIGAPP Applied Computing Review
7	ACM SIGARCH Computer Architecture News
8	ACM SIGART Bulletin
9	ACM SIGAda Ada Letters
10	ACM SIGBED Review
11	ACM SIGBIO Newsletter
12	ACM SIGCAPH Computers and the Physically Handicapped
13	ACM SIGCAS Computers and Society
14	ACM SIGCHI Bulletin
15	ACM SIGCHI Bulletin - a supplement to interactions
16	ACM SIGCOMM Computer Communication Review
17	ACM SIGCPR Computer Personnel
18	ACM SIGCSE Bulletin
19	ACM SIGCSIM Installation Management Review
20	ACM SIGCUE Outlook
21	ACM SIGDA Newsletter
22	ACM SIGDOC Asterisk Journal of Computer Documentation
23	ACM SIGEVOlution
24	ACM SIGFORTH Newsletter
25	ACM SIGGRAPH Computer Graphics
26	ACM SIGGROUP Bulletin
27	ACM SIGICE Bulletin
28	ACM SIGIR Forum
29	ACM SIGITE Newsletter
30	ACM SIGKDD Explorations Newsletter

Figure 3.18: Newsletters Excel Sheet

Figure 3.19 shows Proceedings worksheet. We have imported the Proceedings table from ACM Web site, and it contains the following columns:

1. *Proceeding_No*: This attribute indicates a unique number for each Proceeding, and will be the table's Primary key when it will be imported to the database.
2. *Proceeding*: This attribute contains the Proceeding name and description.



Microsoft Excel - ACM

File Edit View Insert Format Tools Data Window Help Adobe PDF

Type a question for help

Arial 10 100%

Create PDF

B76 FMSP: Formal Methods in Software Practice

	A	B	C	D	E	F
1	Proceeding_No	Proceeding				
2	1	A-MOST: Workshop on Advances in Model-Based Testing				
3	2	ACM Policy: ACM Policy				
4	3	ACM-SE: ACM Southeast Regional Conference				
5	4	ACM/CSC-ER: ACM Annual Conference/Annual Meeting				
6	5	AFRIGRAPH: Computer graphics, virtual reality, visualisation and interaction in Africa				
7	6	AGENTS: International Conference on Autonomous Agents				
8	7	AICPS: ACM International Conference Proceeding Series				
9	8	ANCS: Symposium On Architecture For Networking And Communications Systems				
10	9	ANNA: Analysis of Neural Net Applications Conference				
11	10	ANSS: Annual Simulation Symposium				
12	11	AOSD: Aspect-oriented software development				
13	12	APL: International Conference on APL				
14	13	ASE: Automated Software Engineering				
15	14	ASPDAC: with EDA Technofair Design Automation Conference Asia and South Pacific				
16	15	ASPLOS: Architectural Support for Programming Languages and Operating Systems				
17	16	ASSETS: ACM SIGACCESS Conference on Assistive Technologies				
18	17	AVI: AVI				
19	18	C&C: Creativity and Cognition				
20	19	CASES: International Conference on Compilers, Architecture and Synthesis for Embedded Systems				
21	20	CAW: Computer Architecture Workshop				
22	21	CC: Critical Computing				
23	22	CCS: Conference on Computer and Communications Security				
24	23	CCSC: Consortium for Computing Sciences in Colleges				
25	24	CCU: Contemporary Computing in Ukraine				
26	25	CF: Conference On Computing Frontiers				
27	26	CFP: Computers, Freedom and Privacy				
28	27	CGO: Code Generation and Optimization				
29	28	CHI: Conference on Human Factors in Computing Systems				
30	29	CHIMIT: Computer Human Interaction for the Management of Information Technology				
31	30	CIKM: Conference on Information and Knowledge Management				

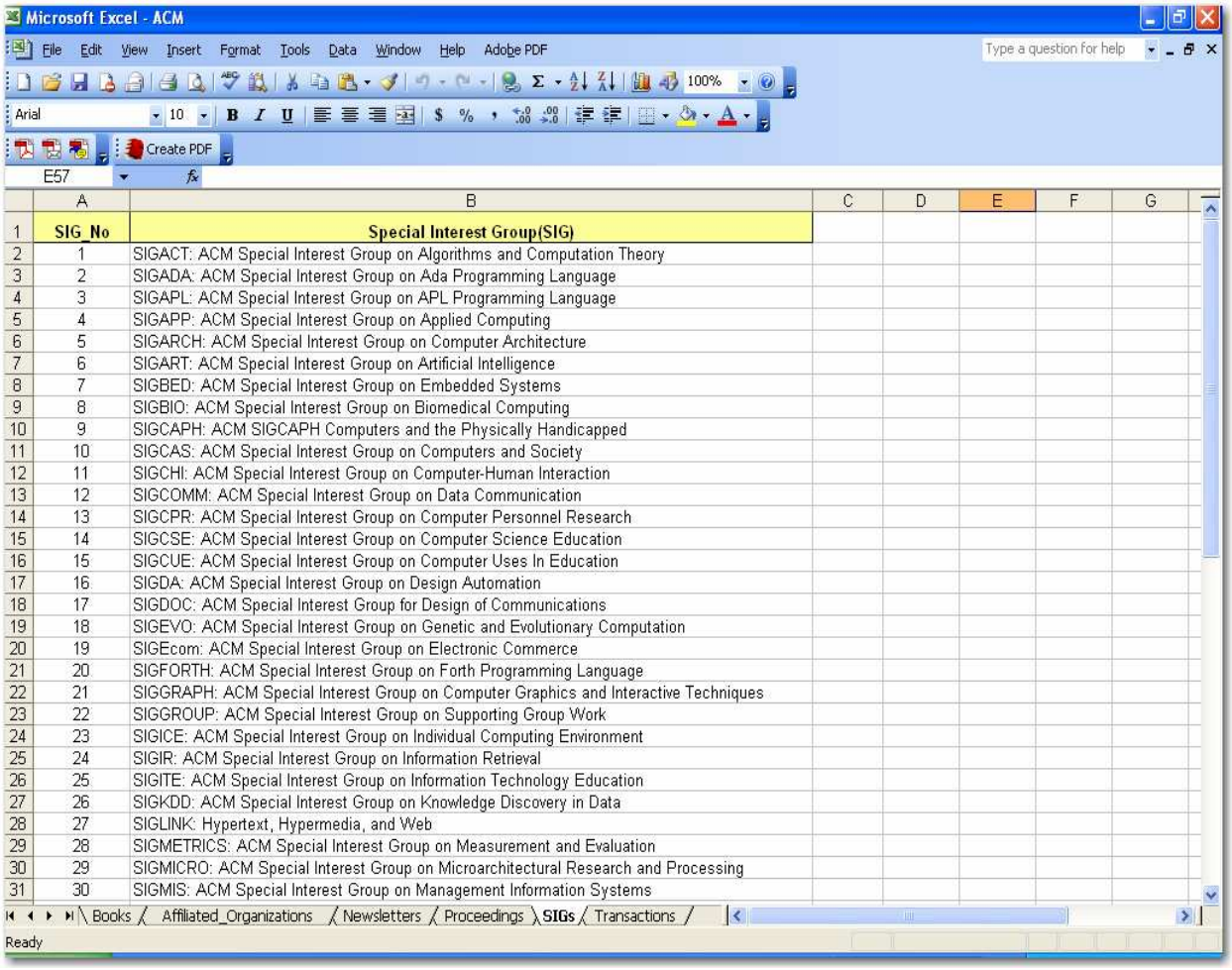
Books Affiliated_Organizations Newsletters **Proceedings** SIGs Transactions

Ready

Figure 3.19: Proceedings Excel Sheet

Figure 3.20 shows Special Interest Groups (SIGs) worksheet. We have imported the SIGs table from ACM Web site, and it contains the following columns:

1. *SIG_No*: This attribute indicates a unique number for each Special Interest Group, and will be the table's Primary key when it will be imported to the database.
2. *Special Interest Group (SIG):* This attribute contains the Special Interest Group name.

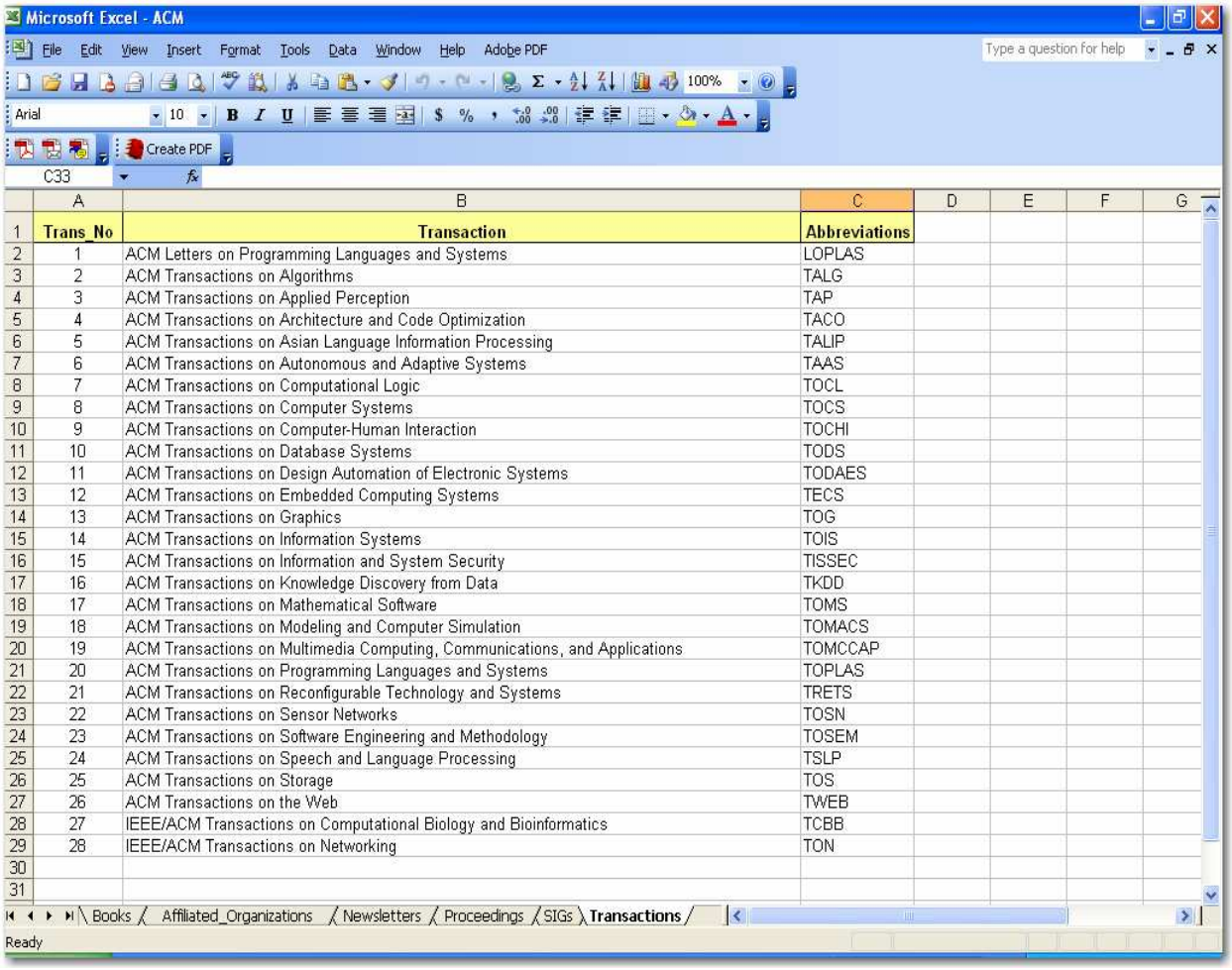


	A	B
1	SIG_No	Special Interest Group(SIG)
2	1	SIGACT: ACM Special Interest Group on Algorithms and Computation Theory
3	2	SIGADA: ACM Special Interest Group on Ada Programming Language
4	3	SIGAPL: ACM Special Interest Group on APL Programming Language
5	4	SIGAPP: ACM Special Interest Group on Applied Computing
6	5	SIGARCH: ACM Special Interest Group on Computer Architecture
7	6	SIGART: ACM Special Interest Group on Artificial Intelligence
8	7	SIGBED: ACM Special Interest Group on Embedded Systems
9	8	SIGBIO: ACM Special Interest Group on Biomedical Computing
10	9	SIGCAPH: ACM SIGCAPH Computers and the Physically Handicapped
11	10	SIGCAS: ACM Special Interest Group on Computers and Society
12	11	SIGCHI: ACM Special Interest Group on Computer-Human Interaction
13	12	SIGCOMM: ACM Special Interest Group on Data Communication
14	13	SIGCPR: ACM Special Interest Group on Computer Personnel Research
15	14	SIGCSE: ACM Special Interest Group on Computer Science Education
16	15	SIGCUE: ACM Special Interest Group on Computer Uses In Education
17	16	SIGDA: ACM Special Interest Group on Design Automation
18	17	SIGDOC: ACM Special Interest Group for Design of Communications
19	18	SIGEVO: ACM Special Interest Group on Genetic and Evolutionary Computation
20	19	SIGEcom: ACM Special Interest Group on Electronic Commerce
21	20	SIGFORTH: ACM Special Interest Group on Forth Programming Language
22	21	SIGGRAPH: ACM Special Interest Group on Computer Graphics and Interactive Techniques
23	22	SIGGROUP: ACM Special Interest Group on Supporting Group Work
24	23	SIGICE: ACM Special Interest Group on Individual Computing Environment
25	24	SIGIR: ACM Special Interest Group on Information Retrieval
26	25	SIGITE: ACM Special Interest Group on Information Technology Education
27	26	SIGKDD: ACM Special Interest Group on Knowledge Discovery in Data
28	27	SIGLINK: Hypertext, Hypermedia, and Web
29	28	SIGMETRICS: ACM Special Interest Group on Measurement and Evaluation
30	29	SIGMICRO: ACM Special Interest Group on Microarchitectural Research and Processing
31	30	SIGMIS: ACM Special Interest Group on Management Information Systems

Figure 3.20: Special Interest Groups (SIGs) Excel Sheet

Figure 3.21 shows the last sheet, whish is Transactions worksheet. We have imported the Transactions table from ACM Web site, and it contains the following columns:

1. *Trans_No*: This attribute indicates a unique number for each Transaction, and will be the table's Primary key when it will be imported to the database.
2. *Transaction*: This attribute contains the Transaction name and description.
3. *Abbreviation*: This attribute indicates an abbreviation for the Transaction name.



Microsoft Excel - ACM

Trans_No	Transaction	Abbreviations
1	ACM Letters on Programming Languages and Systems	LOPLAS
2	ACM Transactions on Algorithms	TALG
3	ACM Transactions on Applied Perception	TAP
4	ACM Transactions on Architecture and Code Optimization	TACO
5	ACM Transactions on Asian Language Information Processing	TALIP
6	ACM Transactions on Autonomous and Adaptive Systems	TAAS
7	ACM Transactions on Computational Logic	TOCL
8	ACM Transactions on Computer Systems	TOCS
9	ACM Transactions on Computer-Human Interaction	TOCHI
10	ACM Transactions on Database Systems	TODS
11	ACM Transactions on Design Automation of Electronic Systems	TODAES
12	ACM Transactions on Embedded Computing Systems	TECS
13	ACM Transactions on Graphics	TOG
14	ACM Transactions on Information Systems	TOIS
15	ACM Transactions on Information and System Security	TISSEC
16	ACM Transactions on Knowledge Discovery from Data	TKDD
17	ACM Transactions on Mathematical Software	TOMS
18	ACM Transactions on Modeling and Computer Simulation	TOMACS
19	ACM Transactions on Multimedia Computing, Communications, and Applications	TOMCCAP
20	ACM Transactions on Programming Languages and Systems	TOPLAS
21	ACM Transactions on Reconfigurable Technology and Systems	TRETS
22	ACM Transactions on Sensor Networks	TOSN
23	ACM Transactions on Software Engineering and Methodology	TOSEM
24	ACM Transactions on Speech and Language Processing	TSLP
25	ACM Transactions on Storage	TOS
26	ACM Transactions on the Web	TWEB
27	IEEE/ACM Transactions on Computational Biology and Bioinformatics	TCBB
28	IEEE/ACM Transactions on Networking	TON

Books | Affiliated_Organizations | Newsletters | Proceedings | SIGs | Transactions

Ready

Figure 3.21: Transactions Excel Sheet

3.4 IMPORTING DATA FROM MICROSOFT EXCEL TO ORACLE DATABASE

After the excel tables were formatted and organized to the final format we wanted, now the next step is to import the data to our database. The database we have chosen is Oracle Database 10g, and the importing process will be done using the tool 'TOAD for Oracle'. This process will be shown and discussed in this section.

First of all, after we open Toad, we must choose the database we are going to import the data to, which is in our case named 'orcl'. And we must connect to the database using a specific user name and password.

Before starting the import process, we must create the tables we want to import the data to in the database and their attributes. To create the tables, the commands will be written in the editor window as shown in **Figure 3.22**. And then executed by clicking the button *Execute statement*, or by choosing Editor | Execute statement.

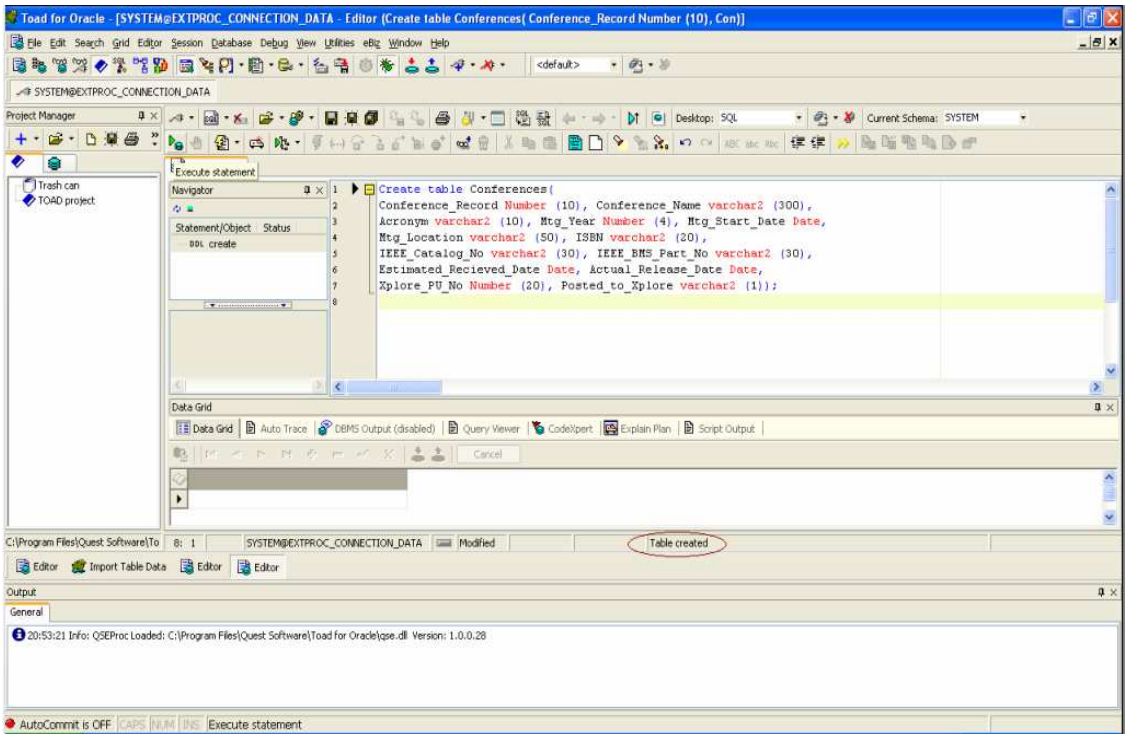


Figure 3.22: Execute "Create table" Command in Toad

We have created a table in the database for every excel sheet, and it is important that the attributes of each table created in the database match the heading of each column in the excel sheet. And here are all the commands we have written and executed, to create our 12 tables in oracle database:

```
1) Create table Conferences(
   Conference_Record Number (10), Conference_Name varchar2 (300),
   Acronym varchar2 (10), Mtg_Year Number (4), Mtg_Start_Date Date,
   Mtg_Location varchar2 (50), ISBN varchar2 (20),
   IEEE_Catalog_No varchar2 (30), IEEE_BMS_Part_No varchar2 (30),
   Estimated_Recieved_Date Date, Actual_Release_Date Date,
   Xplore_PU_No Number (20), Posted_to_Xplore varchar2 (1));

2) Create table Journals(
   Journal_No Number (3), Journal varchar2 (200),
```

```
   Abbreviation varchar2 (10), Source varchar2 (10));

3) Create table Magazines(
   Magazine_No Number (2), Magazine varchar2 (500),
   Abbreviation varchar2 (10), Source varchar2 (10));

4) Create table Educational_Courses(
   Course_ID Number (3), Course_Title varchar2 (200),
   Course_Instructors varchar2 (200), Sponsored_by varchar2 (200),
   Publication_Date Date, Course_Summary varchar2 (600));

5) Create table Articles(
   Article_No Number (3), Authors varchar2 (200),
   Article_Title varchar2 (200), Journal varchar2 (200),
   Article_Year Number (4));

6) Create table Citations(
   Citation_No Number (3), Authors varchar2 (200),
   Citation varchar2 (200), Journal varchar2 (200),
   Citation_Year Number (4));

7) Create table Books(
   Book_ID Number (4), Main_Category varchar2 (100),
   Sub_Category varchar2 (100), Book_Title varchar2 (300),
   Edition varchar2 (30));

8) Create table Affiliated_Organizations(
   Org_NO Number (3), AFF_Organization varchar2 (200),
   Publisher varchar2 (100), Abbreviation varchar2 (10));

9) Create table Newsletters(
   Newsletter_No Number (3), Newsletter varchar2 (200));

10) Create table Proceedings(
    Proceeding_No Number (3), Proceeding varchar2 (200));

11) Create table Special_Interest_Groups(
    SIG_No Number (3), SIG varchar2 (200));

12) Create table Transactions(
    Transaction_No Number (3), Transaction_Title varchar2 (200),
    Abbreviation varchar (10));
```

After all tables were created successfully, the next step is to import the data from the excel sheets to the tables. This is done through the following steps:

1. Choosing Database | Import | Import table data to open the window shown in **Figure 3.23**.

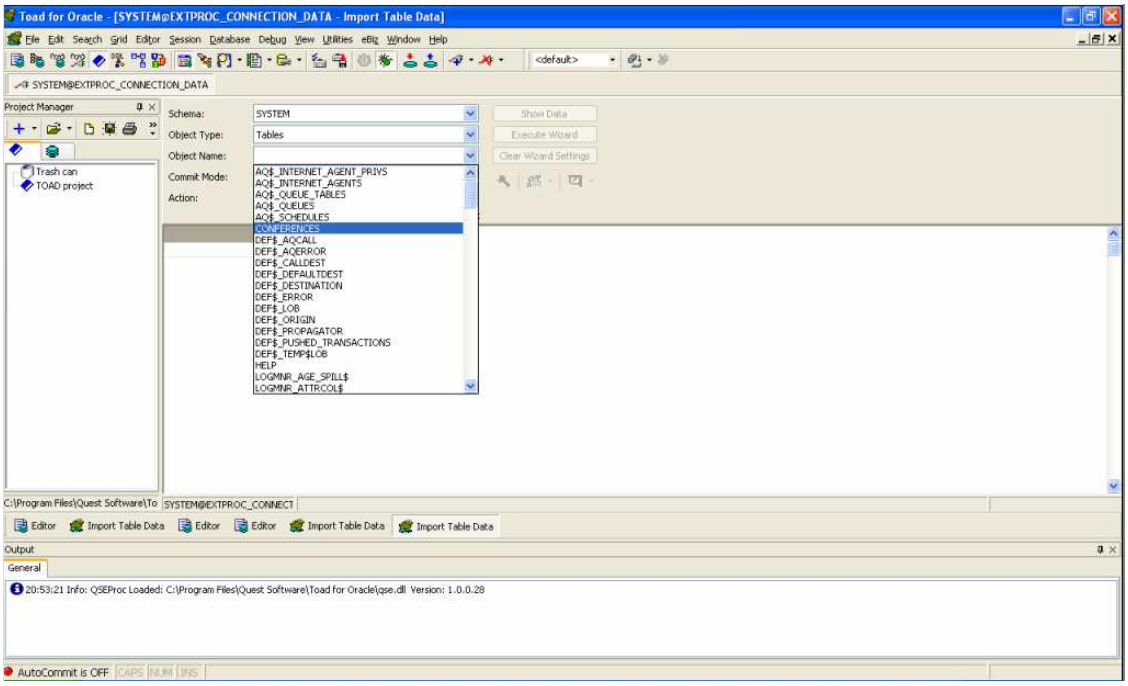


Figure 3.23: Import Table Data

2. Then we select a table name from the *Object Name* list, which is one of the tables we have created in the database. For this example, we are going to select the table "Conferences" to explain the import process.
3. And then, we select the option 'One commit after all records' from the *Commit Mode* list, to make sure that after all records imported to the table, data will be committed.
4. After that, we click the *Show Data* button that will show any data was already in the table, if any, and then click the *Execute Wizard* button to open the Import Wizard window, and then select the table type: Excel file(*.xls) as shown in **Figure 3.24**.

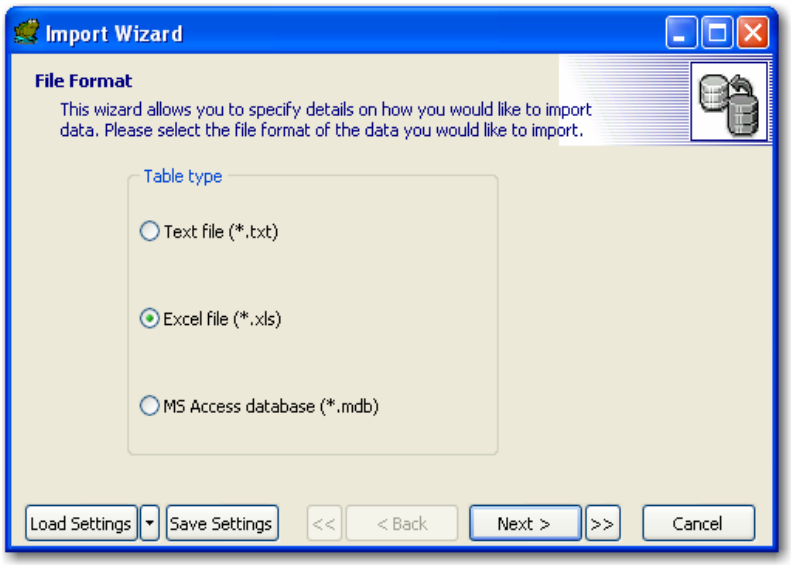


Figure 3.24: Import Wizard Window

5. Then, we choose an excel file that contains the data we would like to import, which is related to the table “Conferences” we want to import the data to, as shown in **Figure 3.25**.

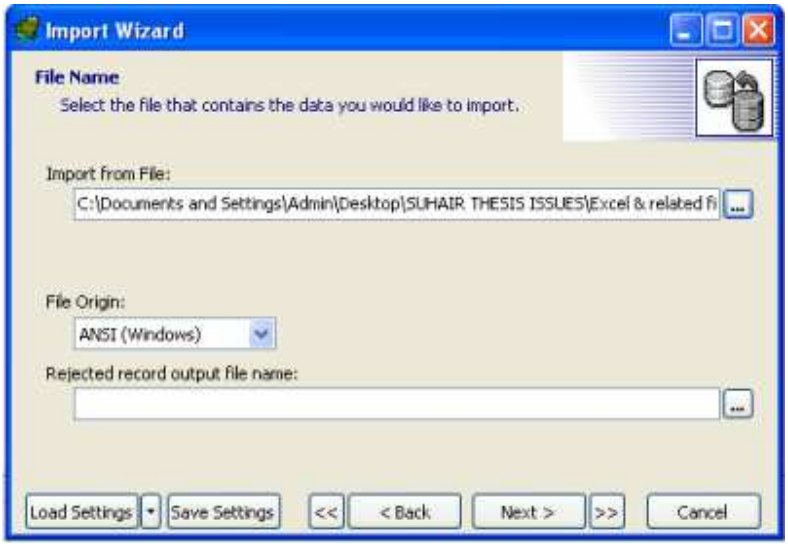


Figure 3.25: Import Wizard Window

6. After that, we must define additional options and formats for the data file, like dates, times and numbers. **Figure 3.26** shows the data formats.

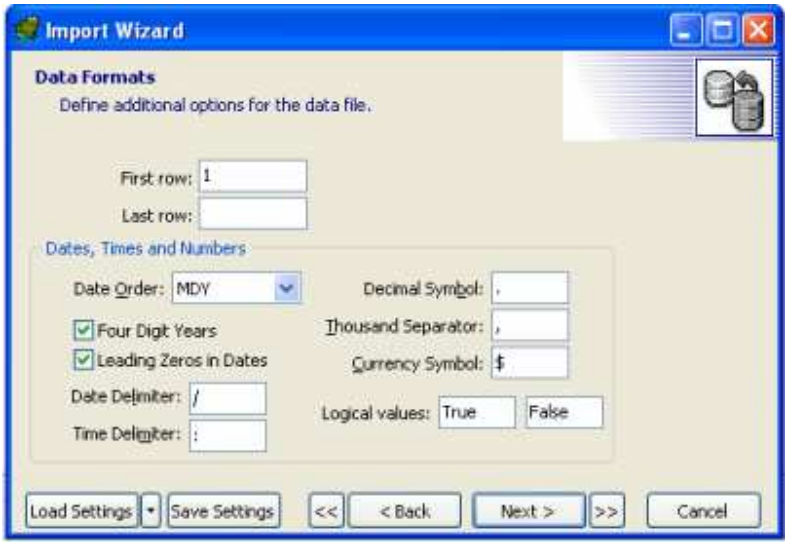


Figure 3.26: Import Wizard Window

7. Then, the import wizard will make some guesses about where our field breaks occur in the file preview. And we could also make our own adjustments, because sometimes the guesses won't be correct. We must choose the excel sheet we want to import data from, then map the columns names to the table attributes as shown in **Figure 3.27**.

Figure 3.27: Import Wizard Window

8. After that, we set the fields mappings to specify the correspondence between fields in the source and destination files, and also choose the field that will be the primary key for the table, which is in our example, the 'conference_record'. **Figure 3.28** shows this process.

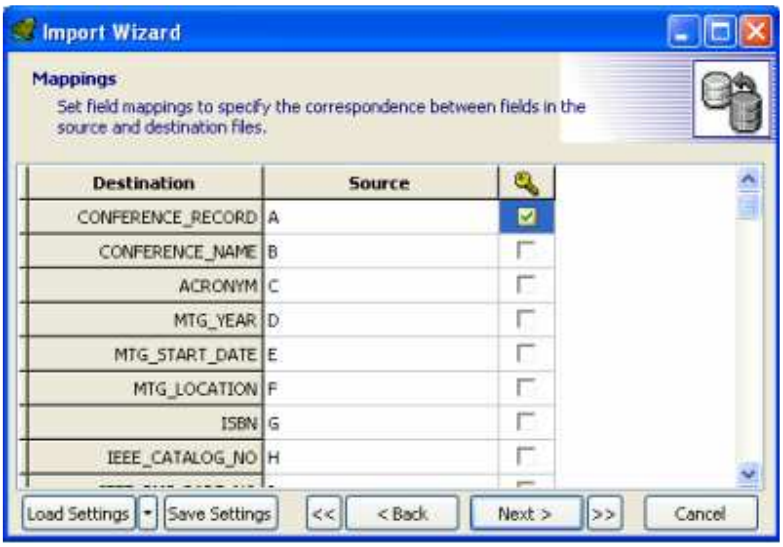


Figure 3.28: Import Wizard Window

9. Then, the data will be previewed, to check it and make sure of its correctness before importing. Also, we click the button 'Size Cols to Data' to size the table columns to the data we are going to import as shown in **Figure 3.29**.

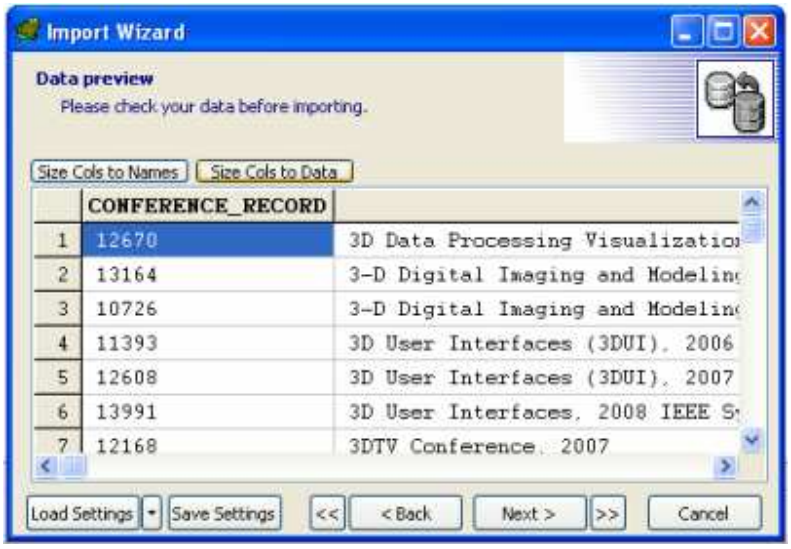


Figure 3.29: Import Wizard Window

10. Finally, a summary will be previewed to check that all records will be added to the destination table. Then, we click the 'Execute' Button to begin the import process as shown in **Figure 3.30**.

Figure 3.30: Import Wizard Window

11. After the import process is done, a result window shown in **Figure 3.31** will present the number of rows processed, added, deleted and updated. And also, will show the total errors if any.

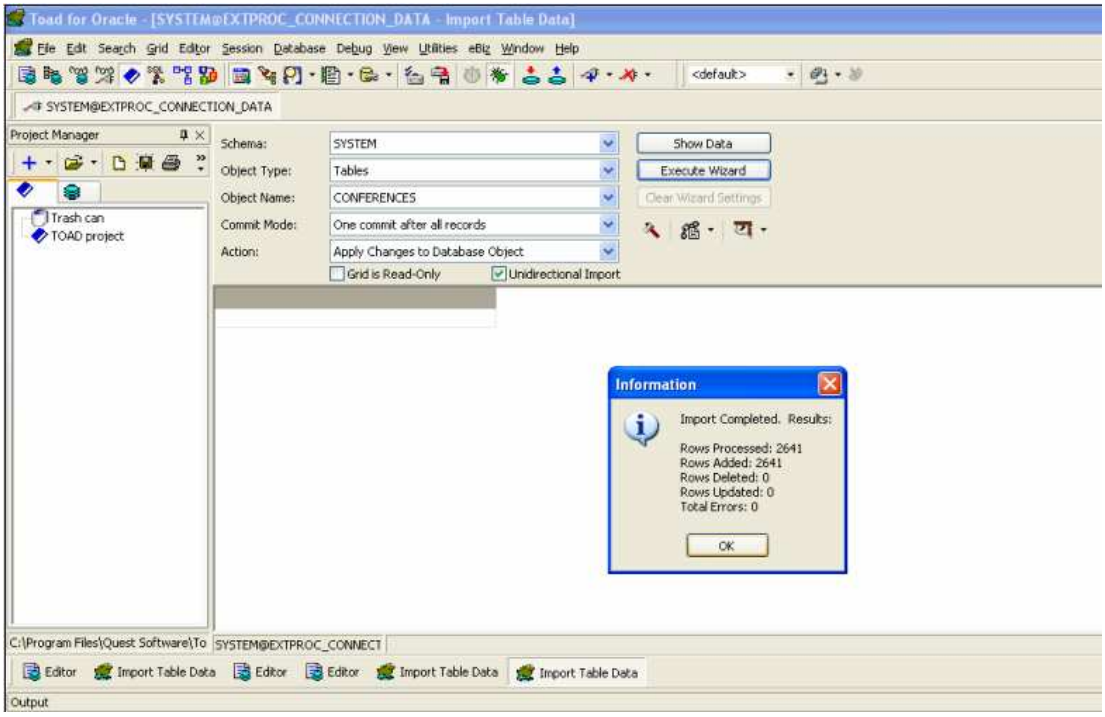


Figure 3.31: Result Window after the Import Data Process is done

12. The final result and the table data will then be previewed after the import process is completed successfully. The data imported to the table "Conferences" is shown in **Figure 3.32** after the import table data process is successfully done.

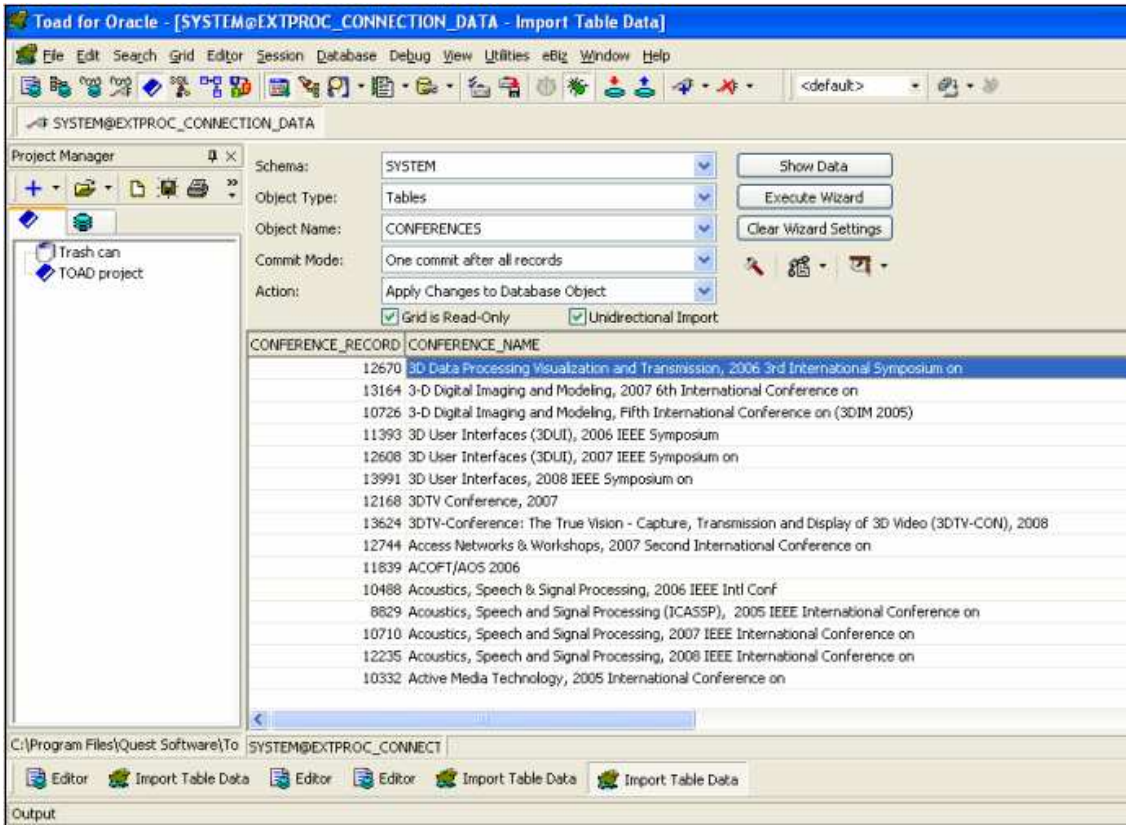


Figure 3.32: End of Import Table Data Process

We have done the previous process to all the 12 tables we have created in the database: Conferences, Journals, Magazines, Educational_Courses, Articles, Citations, Books, Affiliated_Organizations, Newsletters, Proceedings, Special_Interest_Groups and Transactions. Now, our database is successfully created and is ready to test our search generic menu-based interface on, as will be shown in details in the next chapter.

CHAPTER 4

GENERIC MENU-BASED INTERFACE (GMBI) DEVELOPMENT

After creating the tables in the database, and then importing the data into the tables, the next and most important step now is to develop our Generic Menu-Based Interface (GMBI), which the user will interact directly with to enter search queries and make information retrieval. **Figure 4.1** presents this phase of our work that we have described and discussed in this chapter.

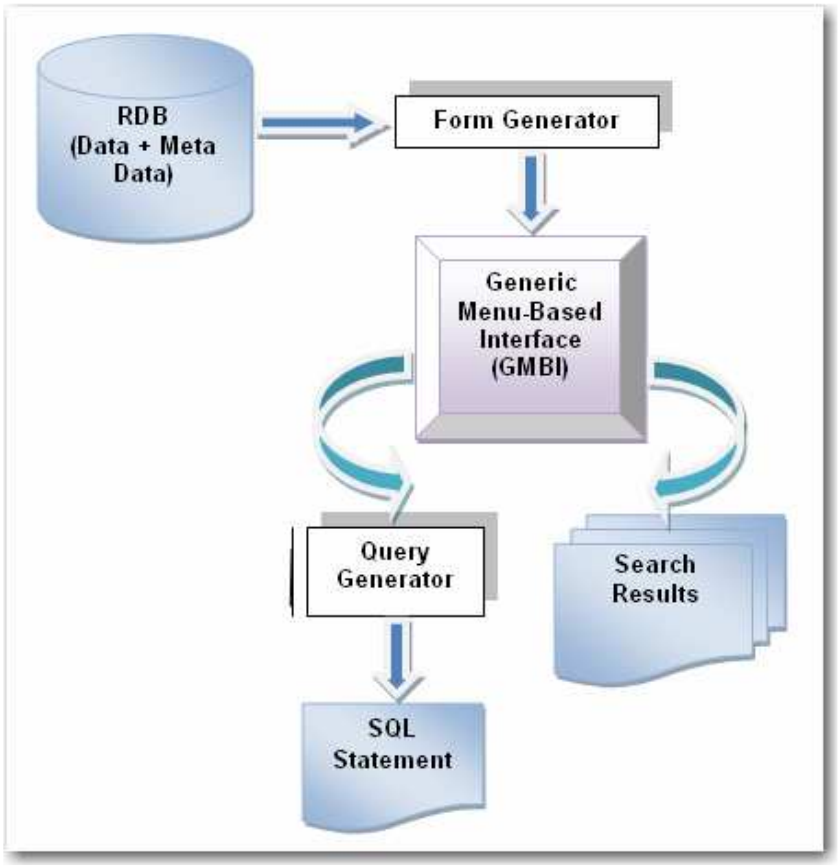


Figure 4.1: Development Process

4.1 OVERVIEW OF THE DATABASE TABLES

In this section, we have viewed the tables created in the database, listing their attributes and the attributes' data types. **Table 4.1** shows table 'Conferences' with its attributes: Conference_Record (Primary Key), Conference_Name, Acronym, Mtg_Year, Mtg_Start Date, Mtg_Location, ISBN, IEEE_Catalog_No, IEEE_BMS_Part_No, Estimated_Received_Date, Actual_Release_Date, Xplore_PU_No, and Posted_to_Xplore.

Table 4.1: Conferences

CONFERENCES	
Attributes	**Data Types**
Conference_Record	**Number (5)**
Conference_Name	**Varchar2 (300)**
Acronym	**Varchar2 (10)**
Mtg_Year	**Number(4)**
Mtg_Start_Date	**Date**
Mtg_Location	**Varchar2 (50)**
ISBN	**Varchar2 (20)**
IEEE_Catalog_No	**Varchar2 (30)**
IEEE_BMS_Part_No	**Varchar2 (30)**
Estimated_Received_Date	**Date**
Actual_Release_Date	**Date**
Xplore_PU_No	**Varchar2 (20)**
Posted_to_Xplore	**Varchar2 (1)**

Table 4.2 describes table 'Journals' which consists of the attributes: Journal_No (Primary Key), Journal, Abbreviation, and Source.

Table 4.2: Journals

JOURNALS	
Attributes	**Data Types**
Journal_No	**Number (3)**
Journal	**Varchar2 (200)**
Abbreviation	**Varchar2 (10)**
Source	**Varchar2 (10)**

Table 4.3 presents table 'Magazines' which consists of the attributes: Magazine_No (Primary Key), Magazine, Abbreviation, and Source.

Table 4.3: Magazines

MAGAZINES	
Attributes	**Data Types**
Magazine_No	**Number (2)**
Magazine	**Varchar2 (500)**
Abbreviation	**Varchar2 (10)**
Source	**Varchar2 (10)**

Table 4.4 shows table 'Educational_Courses' which consists of the following attributes: Course_ID (Primary Key), Course_Title, Course_Instructors, Sponsered_By, Publication_Date, and Course_Summary.

Table 4.4: Educational_Courses

EDUCATIONAL_COURSES	
Attributes	**Data Types**
Course_ID	**Number (3)**
Course_Title	**Varchar2 (200)**
Course_Instructors	**Varchar2 (200)**
Sponsered_By	**Varchar2 (200)**
Publication_Date	**Date**
Course_Summary	**Varchar2 (600)**

Table 4.5 describes table 'Articles' which consists of the following attributes: Article_No (Primary Key), Article_Title, Authors, Journal, and Article_Year.

Table 4.5: Articles

ARTICLES	
Attributes	**Data Types**
Article_No	**Number (3)**
Article_Title	**Varchar2 (200)**
Authors	**Varchar2 (200)**
Journal	**Varchar2 (200)**
Article_Year	**Number (4)**

Table 4.6 describes table 'Citations' which consists of the following attributes: Citation_No (Primary Key), Citation_Title, Authors, Journal, and Citation_Year.

Table 4.6: Citations

CITATIONS	
Attributes	**Data Types**
Citation_No	**Number (3)**
Citation_Title	**Varchar2 (200)**
Authors	**Varchar2 (200)**
Journal	**Varchar2 (200)**
Citation_Year	**Number (4)**

Table 4.7 presents table 'Books' with its attributes: Book_ID (Primary Key), Main_Category, Sub_Category, Book_Title, and Edition.

Table 4.7: Books

BOOKS	
Attributes	**Data Types**
Book_ID	**Number (4)**
Main_Category	**Varchar2 (100)**
Sub_Category	**Varchar2 (100)**
Book_Title	**Varchar2 (300)**
Edition	**Varchar2 (30)**

Table 4.8 shows table 'Affiliated_Organizations' which consists of the attributes: Org_No (Primary Key), Aff_Organization, Publisher, and Abbreviation.

Table 4.8: Affiliated_Organizations

AFFILIATED_ORGANIZATIONS	
Attributes	**Data Types**
Org_No	**Number (3)**
Aff_Organization	**Varchar2 (200)**
Publisher	**Varchar2 (100)**
Abbreviation	**Varchar2 (10)**

Table 4.9 shows table 'Newsletters' which consists of the following attributes: Newsletter_No (Primary Key), and Newsletter.

Table 4.9: Newsletters

NEWSLETTERS	
Attributes	**Data Types**
Newsletter_No **Newsletter**	**Number (3)** **Varchar2 (200)**

Table 4.10 describes table 'Proceedings' which consists of the following attributes: Proceeding_No (Primary Key), and Proceeding.

Table 4.10: Proceedings

PROCEEDINGS	
Attributes	**Data Types**
Proceeding_No **Proceeding**	**Number (3)** **Varchar2 (200)**

Table 4.11 presents table 'Special_Interest_Groups' which consists of the following attributes: SIG_No (Primary Key), and SIG.

Table 4.11: Special Interest Groups

SPECIAL_INTREST_GROUPS	
Attributes	**Data Types**
SIG_No **SIG**	**Number (3)** **Varchar2 (200)**

Table 4.12 describes table 'Transactions' which consists of the following attributes: Transaction_No (Primary Key), Transaction_Title, and Abbreviation.

Table 4.12: Transactions

TRANSACTIONS	
Attributes	**Data Types**
Transaction_No **Transaction_Title** **Abbreviation**	**Number (3)** **Varchar2 (200)** **Varchar2 (10)**

Those tables mentioned above and their data will be analyzed and studied in order to deign our interface. Next section will be devoted to this idea.

4.2 ANALYSIS AND DESIGN OF GMBI

Regardless of the tools and techniques that one uses to develop and build a system, the underlying components of systems development are essentially the same [34]. In our work, we followed the traditional WaterFall model, which consists of the following major phases: Analysis, Design, Implementation, Testing, and Maintenance. The model is described in Appendix B.

The analysis and design phases are complex and critical steps in determining which design, based on systems engineering and technology analysis, meets the user and system requirements [34].

In order to design our generic search Interface, the problem was specified and identified, and we searched through the Internet and WWW about searchers and users requirements and needs. Furthermore, we studied other search forms and interfaces and analyzed them and tried to present something new and more useful for users in our Interface as will be shown in this of this chapter.

While analyzing the Web sites that concern publications and their data, we selected the domain that building and developing our Interface will depend on. And then, we have created the database with its tables that we have described earlier. Eventually, we have collected good and enough information to begin the design phase.

In the design phase, the system specifications are translated into a software representation and architecture. We used Oracle forms builder 10g to design and then implement our Interface. In our work, the design phase consists of two stages: creating the Data blocks and their items, and creating the Canvases.

Our interface consists of 13 data blocks which are created using the Data Block Wizard, 12 of them are database blocks which present the tables created in the database

and their attributes, and the last data block is a control block which presents the search and control process as will be discussed later in this chapter.

To create a database data block using the Data Block Wizard, we first select the type of the data block we would like to create which could be a table or view, or stored procedure. The type of all the data blocks we would like to create is table, because they will be built on our database tables. After that, we enter the table on which we want to base the data block. Then we select the columns that should appear as items in the data block as shown in **Figure 4.2**.

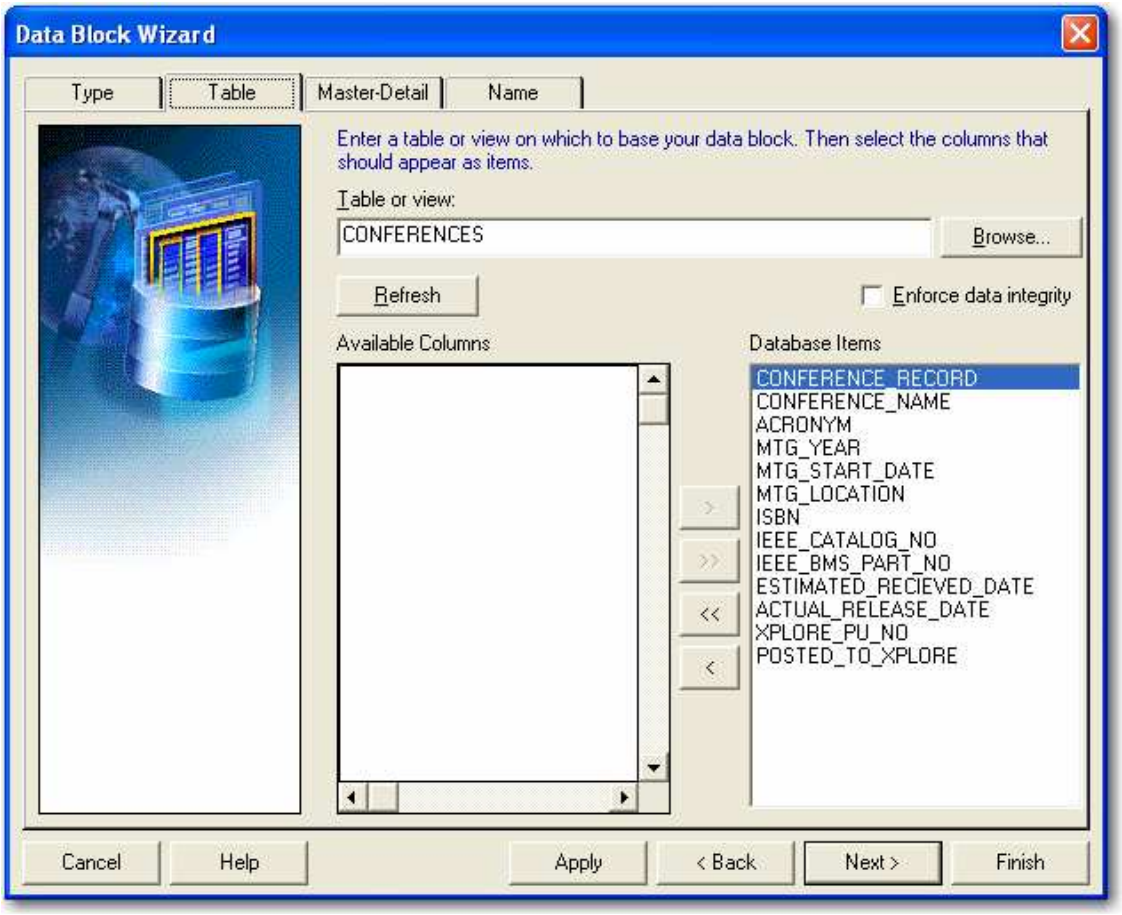


Figure 4.2: Data Block Wizard

Furthermore, in the Data Block Wizard we may optionally create master-detail relationships to other data blocks created in our form. Finally, we enter a name for the data block and it will be successfully created.

The next stage in the design phase is creating the Canvases. We divide our interface into two Canvases and they will be created using the Layout Wizard, although we could also create them manually. But the Layout Wizard allows us to quickly and easily lay out

the items of the data blocks on the Canvas, and will display the items in a frame on the Canvas. First, we select the Canvas type on which we want to layout the data block's items. The Canvas could be Content, Stacked, Vertical Toolbar (VTB), Horizontal Toolbar (HTB), or Tab.

We select our Canvases to be: a Horizontal Toolbar Canvas and a Tab Canvas. When creating the Tab Canvas, we also must select a Tab page on which to layout the data block's items. In our case, we select 12 different Tab pages which represent the data blocks created for the 12 tables in our database.

Then, we select the data block we want and the items that should be displayed in the frame, and select an item type for each. After that, we enter a prompt, height, and width for each item as shown in **Figure 4.3**.

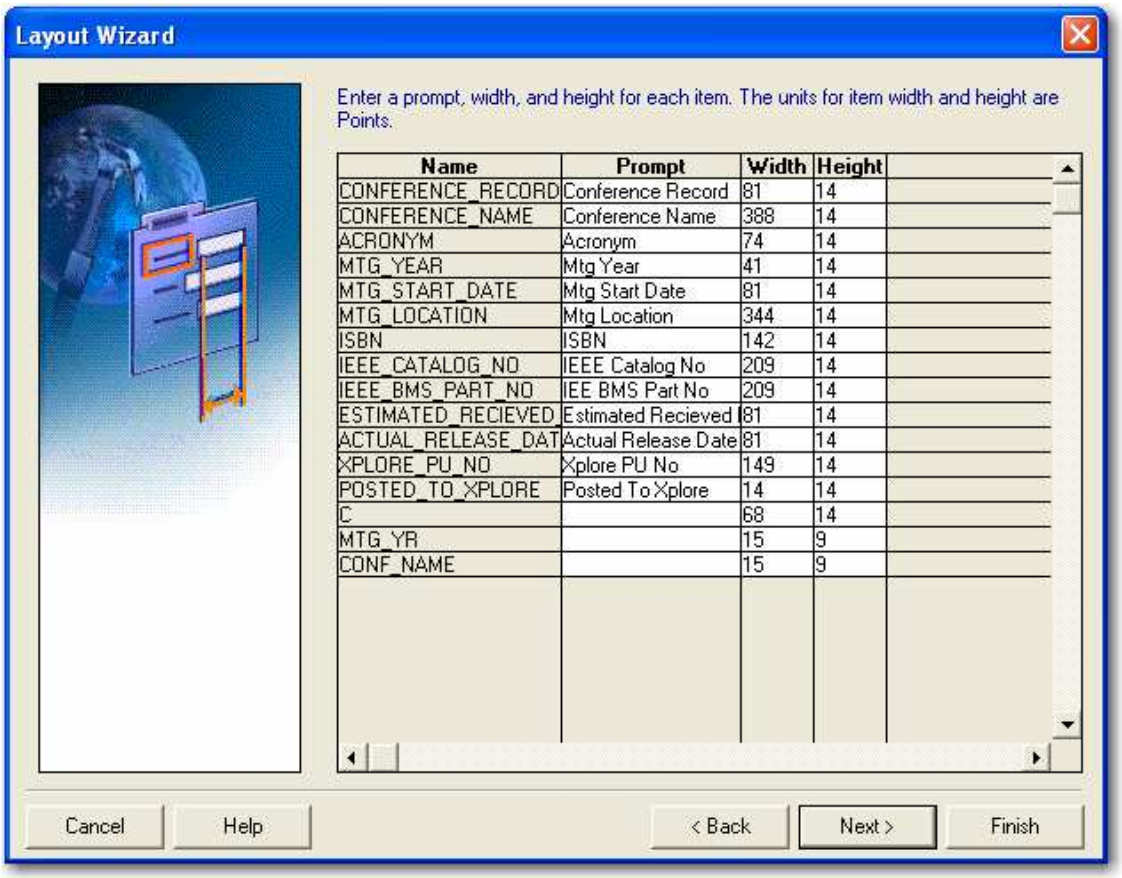


Figure 4.3: Layout Wizard

After that, we select a layout style for our frame to be Form or Tabular. In our case, we select the layout style for our two Canvases to be Form. Finally, we may enter a title for the frame and we must specify the number of database records to be displayed in the frame, as well as the distance between each record. And also, choose whether to display a scrollbar in the frame or not.

The Horizontal Toolbar Canvas shown in **Figure 4.4** is built on the search data block, and contains the following items:

1) **'Search'** Button: When pressing this button, the search process will be executed and done, and the search result will be displayed and presented to the user.
2) **'Show SQL'** Button: When pressing this button, the SQL statement related to the current search process will be viewed in the 'SQL Produced' text item.
3) **'SQL Produced'** Text Item: After pressing the 'Show SQL' button, the SQL statement produced will be viewed in this text item.
4) **'Reset'** Button: When pressing this button, the search process will be reset to let the user begin a new one, and every item will become empty.
5) **'Exit'** Button: This button will let the user exit the search form.

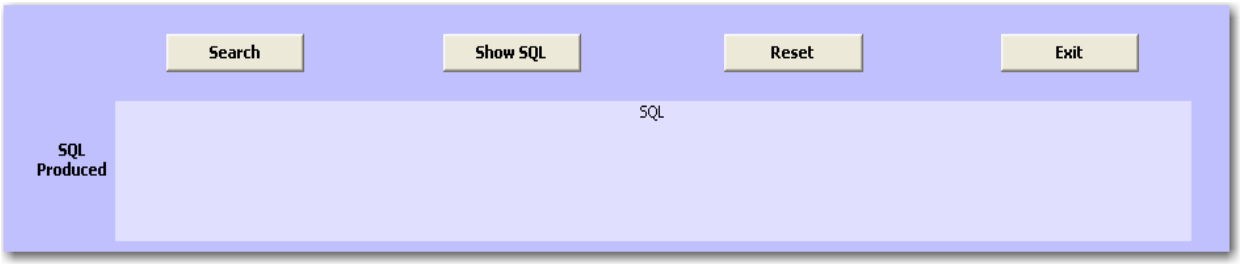


Figure 4.4: The Horizontal Toolbar Canvas and its Items

The Tab Canvas shown in **Figure 4.5** consists of 12 tabs; each is built on one of the data blocks that represent the tables created in the database that were described earlier, and let the search process executes on that table. Each tab contains all the attributes of the table, and allows the search process to be done depending on any of those attributes.

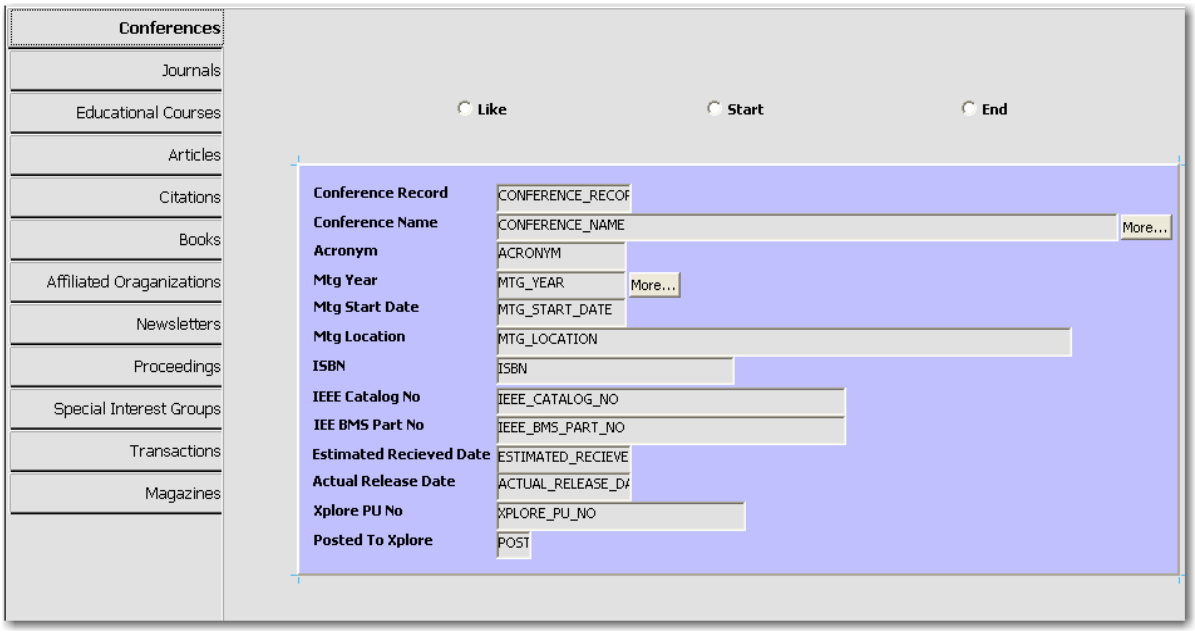


Figure 4.5: The Tab Canvas and its Items

The Tab Canvas also contains a set of radio buttons as follows:

1) **'Like'**: Selected if the user wants to make a search that one of the database items will include the text written by the user in that text item.
2) **'Start'**: Selected if the user wants to make a search that one of the database items will start with the text written by the user in that text item.
3) **'End'**: Selected if the user wants to make a search that one of the database items will end with the text written by the user in that text item.

Furthermore, The Tab Canvas contains buttons labeled **'More…'** related to some of the database items. When selecting one of these buttons, a Menu which contains all the records saved in that item in the database, will pop up and let the user select one of its records and make the search process depending on the record selected, which simplifies the search process a lot for the users.

After the user runs the form, our Generic Menu-Based Interface will look like the screen shot shown in **Figure 4.6**.

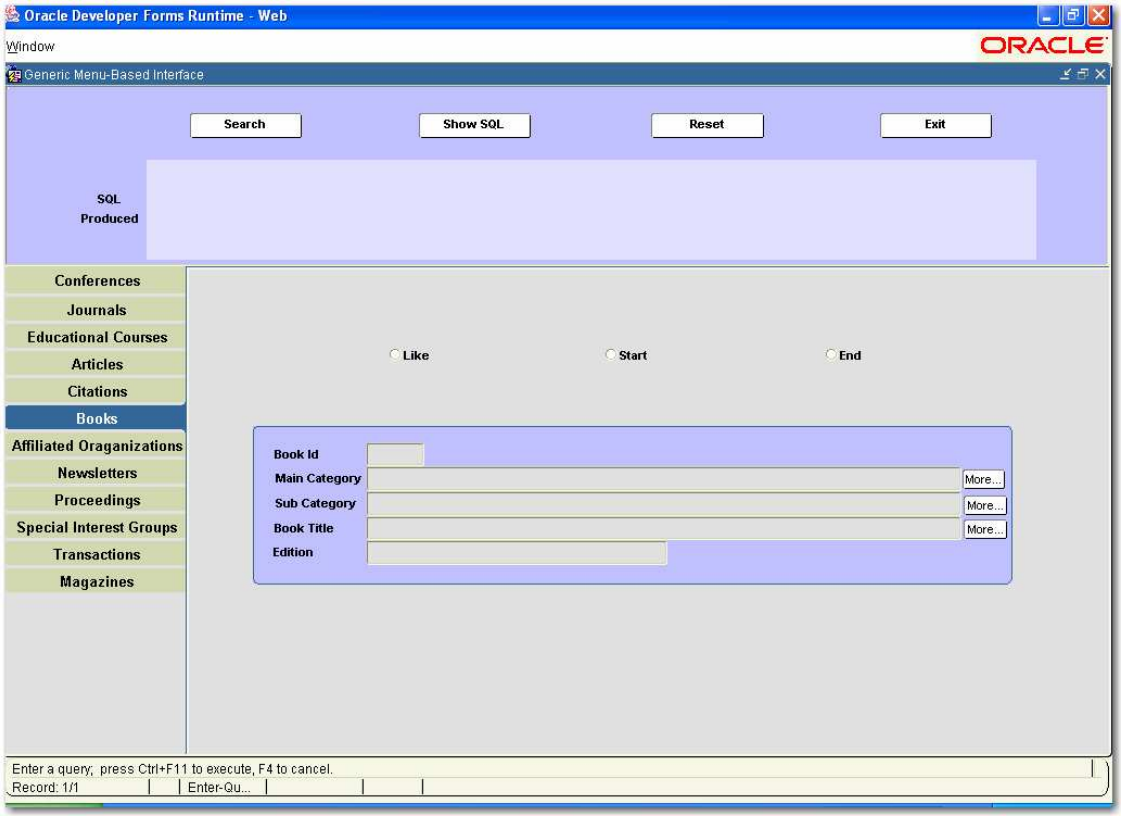


Figure 4.6: Our Generic Menu-Based Interface (GMBI)

After the design phase was completed, our next step will be to begin the implementation and coding phase, and then testing our interface. Next section will be devoted to this idea.

4.3 IMPLEMENTATION OF GMBI

During the implementation phase, the software design is converted into a set of programs or program units. Using Oracle Forms Builder 10g, the Implementation process will be addressed by two stages: building a set of Triggers and writing a PL\SQL code on them, and creating Lists of Values (LOVs) to be our Menus.

Triggers could be built at three levels: Form level, Data block level and Item level. The triggers built at the form level are the most generic. Examples of those triggers that we built during the implementation phase are: PRE-FORM, WHEN-TAB-PAGE-CHANGED, ON-MESSAGE, and WHEN-NEW-FORM-INSTANCE. For example, the code that is written on the trigger WHEN-TAB-PAGE-CHANGED is mentioned below.

```
if :system.tab_new_page = 'CONFERENCES' then
    :global.tab:= 1;

elsif :system.tab_new_page = 'JOURNALS' then
    :global.tab:= 2;

elsif :system.tab_new_page = 'EDUCATIONAL_COURSES' then
    :global.tab:= 3;

elsif :system.tab_new_page = 'ARTICLES' then
    :global.tab:= 4;

elsif :system.tab_new_page = 'CITATIONS' then
    :global.tab:= 5;

elsif :system.tab_new_page = 'BOOKS' then
    :global.tab:= 6;

elsif :system.tab_new_page = 'AFFILIATED_ORGANIZATIONS' then
    :global.tab:= 7;

elsif :system.tab_new_page = 'NEWSLETTERS' then
    :global.tab:= 8;

elsif :system.tab_new_page = 'PROCEEDINGS' then
    :global.tab:= 9;

elsif :system.tab_new_page = 'SPECIAL_INTEREST_GROUPS' then
    :global.tab:= 10;

elsif :system.tab_new_page = 'TRANSACTIONS' then
    :global.tab:= 11;
```

```
elsif :system.tab_new_page = 'MAGAZINES' then
    :global.tab:= 12;
end if;

if :system.mode = 'ENTER-QUERY' then
    exit_form;
end if;

:SQL:= NULL;
go_block (:system.tab_new_page);
enter_query;
```

An example of the triggers built at the data block level is: WHEN-NEW-BLOCK-INSTANCE. The most specific and important triggers are the ones built at the item level. The most commonly used are the trigger WHEN-BUTTON-PRESSED which is built on Push Buttons. The function of the buttons that are labeled 'More…' is to view the list of values of a particular database item as a menu. For example the code written on trigger WHEN-BUTTON-PRESSED that is built on the button CONF_NAME on the data block CONFERENCES to show its LOV is shown below.

```
if show_lov ('CONF_LOV') = true then
    null;
end if;
```

Other buttons that are created on the search block like: Search, Reset, Show SQL, and Exit have different functions. The most critical and important trigger for the purpose of GMBI is built on the Search button. A part of the code written on the trigger WHEN-BUTTON-PRESSED which is built on the Search button is shown below.

```
if :global.tab = 2 then

    if :journals.c = 2 then

        :JOURNALS.JOURNAL_NO:= '%' | |:JOURNALS.JOURNAL_NO| | '%';

        :JOURNALS.JOURNAL:= '%' | |:JOURNALS.JOURNAL| | '%';

        :JOURNALS.ABBREVIATION:= '%' | |:JOURNALS.ABBREVIATION| | '%';

        :JOURNALS.SOURCE:= '%' | |:JOURNALS.SOURCE| | '%';

    elsif :journals.c = 3 then

        :JOURNALS.JOURNAL_NO:= :JOURNALS.JOURNAL_NO| | '%';

        :JOURNALS.JOURNAL_NO:= :JOURNALS.JOURNAL| | '%';

        :JOURNALS.JOURNAL_NO:= :JOURNALS.ABBREVIATION| | '%';
```

```
        :JOURNALS.JOURNAL_NO:= :JOURNALS.SOURCE| | '%';
    elsif :journals.c = 4 then
        :JOURNALS.JOURNAL_NO:= '%' | |:JOURNALS.JOURNAL_NO;
        :JOURNALS.JOURNAL:= '%' | |:JOURNALS.JOURNAL;
        :JOURNALS.ABBREVIATION:= '%' | |:JOURNALS.ABBREVIATION;
        :JOURNALS.SOURCE:= '%'| |:JOURNALS.SOURCE;
        end if;
```

The code sample mentioned above concern the Tab page ‘Journals’ and the search process done on it. A similar code is built for all the other Tab pages in the previous trigger.

Our next stage in the implementation phase is creating the Lists of Values (LOVs). The LOVs is created for selected database items that we want to build a menu on, depending on the records of those items to be shown to the users to choose from during their search.

Creating a LOV consists of two main steps: creating a Record Group which the LOV is getting its data from based on an SQL query, and then building the LOV using the LOV wizard or manually. Another way to create a LOV is building it and its Record Group in parallel using the LOV wizard.

In the LOV wizard, we could modify the LOV’s existing Record Group if was created before, or create a new Record Group. Then we build the SQL query that the Record Group will be based on. We may enter the SQL query directly into an SQL query statement field, or use the Oracle Developer Query Builder to build the query as shown in **Figure 4.7**.

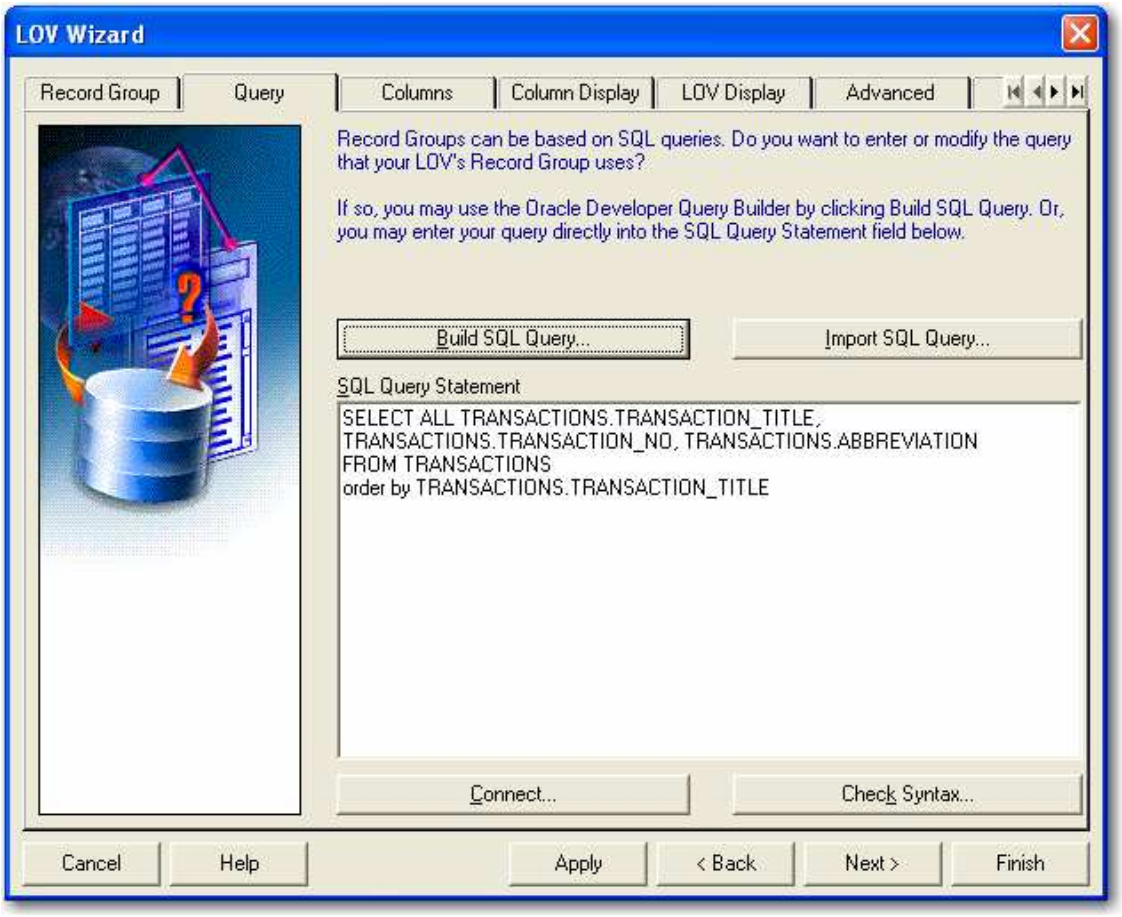


Figure 4.7: LOV Wizard

After that, we determine the Record Group columns that we want to include in the LOV. Then we could specify every LOV column properties, like entering a title, width, and return value for every LOV column. We could also display the title we want in the LOV window and determine its size. We may also modify some advanced properties that affect the behavior of the LOV, like choosing to refresh the Record Group data before displaying the LOV to the user. Finally, we assign the LOV to the items we want to return values to, and then the LOV is successfully created.

By completing all the previous steps and creating all the triggers, LOVs and program units needed for our Generic Menu-Based Interface, the implementation phase is accomplished. In the next section we are going to talk about the testing and maintenance phases.

4.4 TESTING, VALIDATION AND MAINTENANCE OF GMBI

Software validation is intended to show that the system conforms to its specification and meets the expectations of the users [34]. We have tested our system and checked the validation of our generic interface through the three stages of the testing process described in Appendix B. Unit testing involves checking all program units and the triggers that were built one by one and verifying that each unit meets its specification. After that, the individual program units are integrated and tested as a complete system to ensure that the software requirements have been met. The System is tested firstly with simulated test data, and then with real data.

To show how the testing and evaluation phase is accomplished for our system, we have explained this process through the following examples:

Example (1): In this example, the user is looking for a specific Journal, which he knows the abbreviation for. He accesses the 'Journals' tab page and enters the abbreviation for the journal's name which is: "TCBB", in the Abbreviation field as shown in **Figure 4.8**.

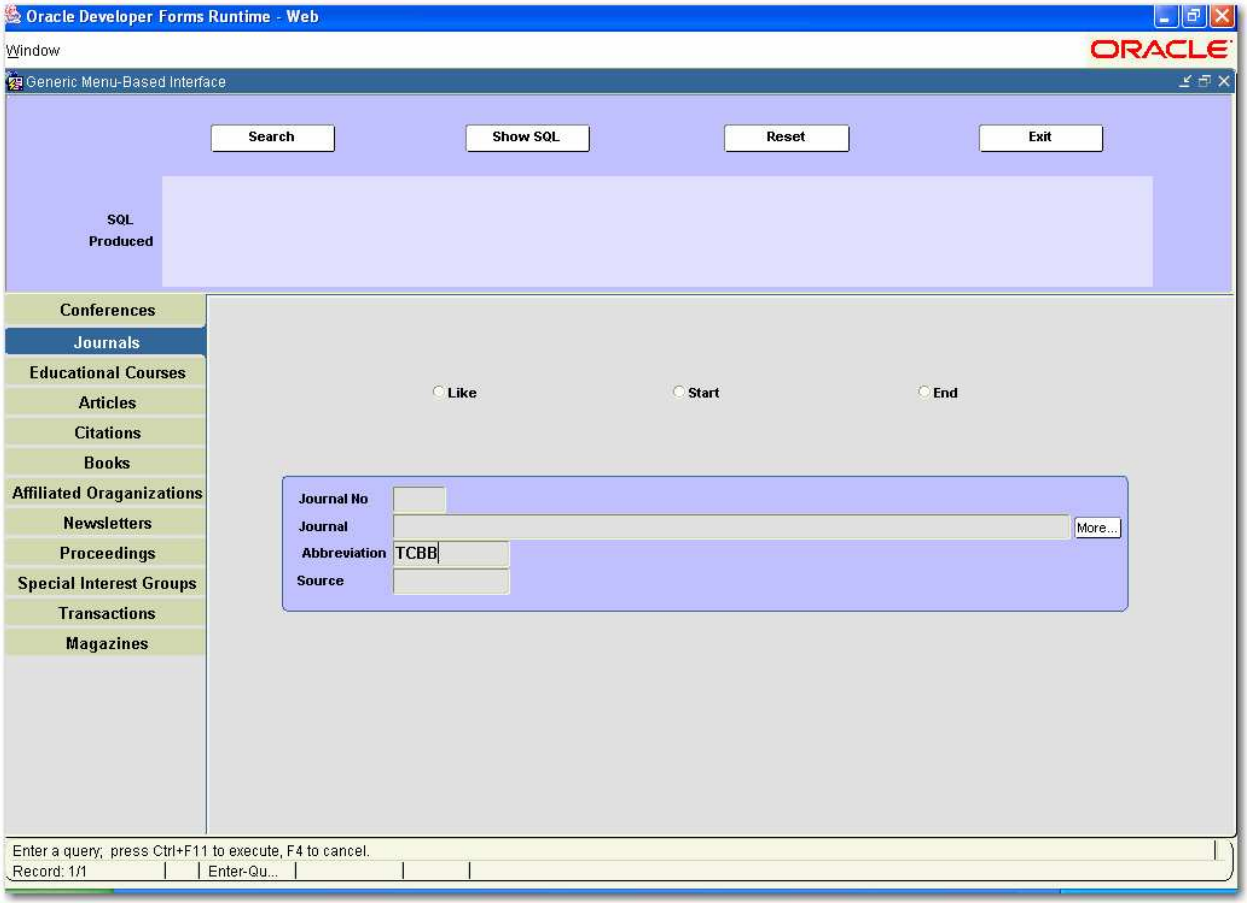


Figure 4.8: Searching for a Specific Journal Depending on the Abbreviation

After that, the searcher presses the ‘Search’ button to view the result of his search, and he could also press the ‘Show SQL’ button to view the SQL statement related to the search process. The search result and the data needed is found and previewed easily and quickly as shown in **Figure 4.9**. And, if the user wants to start a new search, he presses the ‘Reset’ button to clear all the fields and start over again.

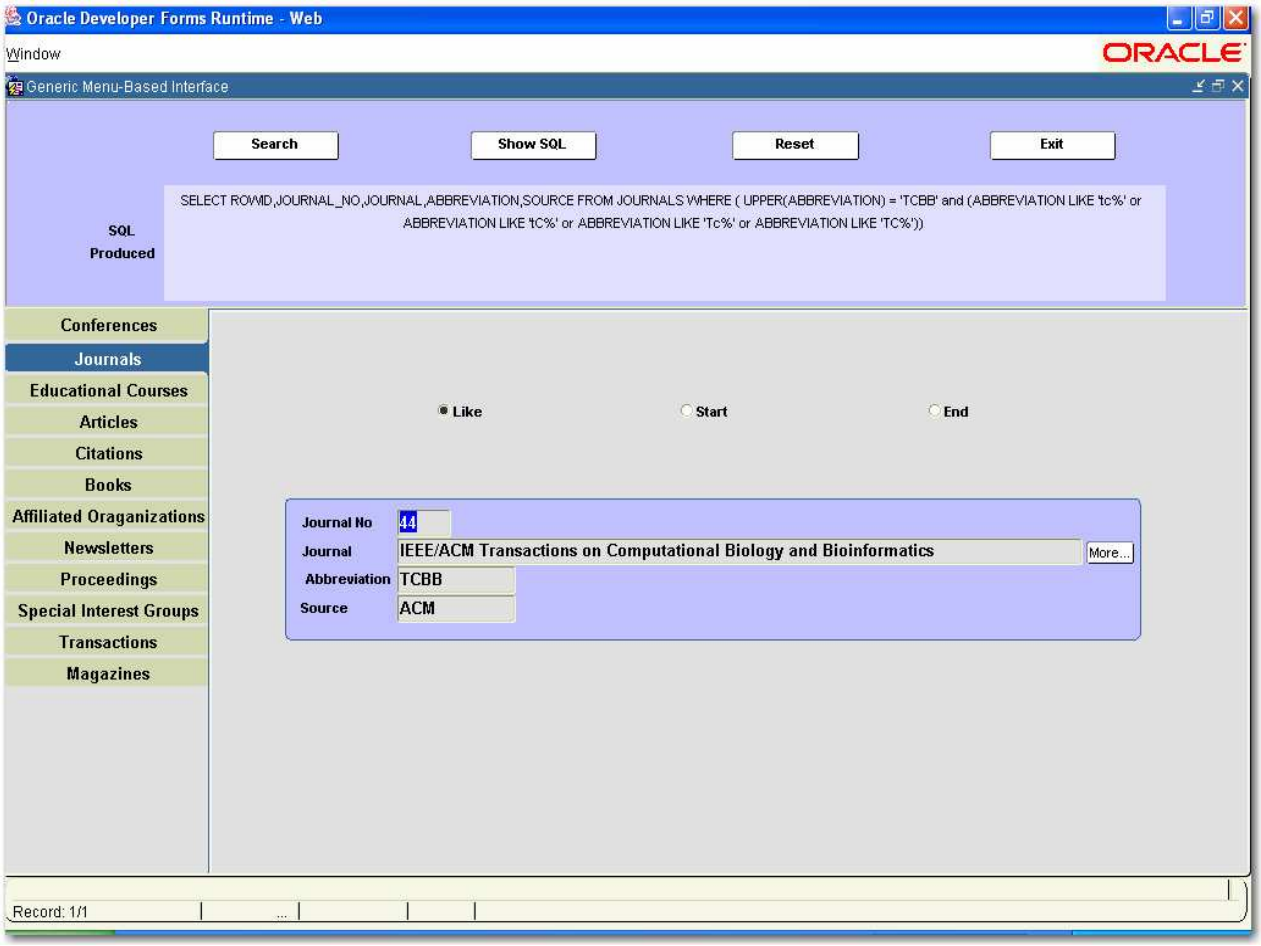


Figure 4.9: The Result of the Search Process Done in Example (1)

Example (2): In this example, the user is looking for all conferences with conference record including number “4”. The user accesses the ‘Conferences’ tab page, and enters “4” in the ‘Conference Record’ field, and then selects the radio button ‘Like’ as shown in **Figure 4.10**.

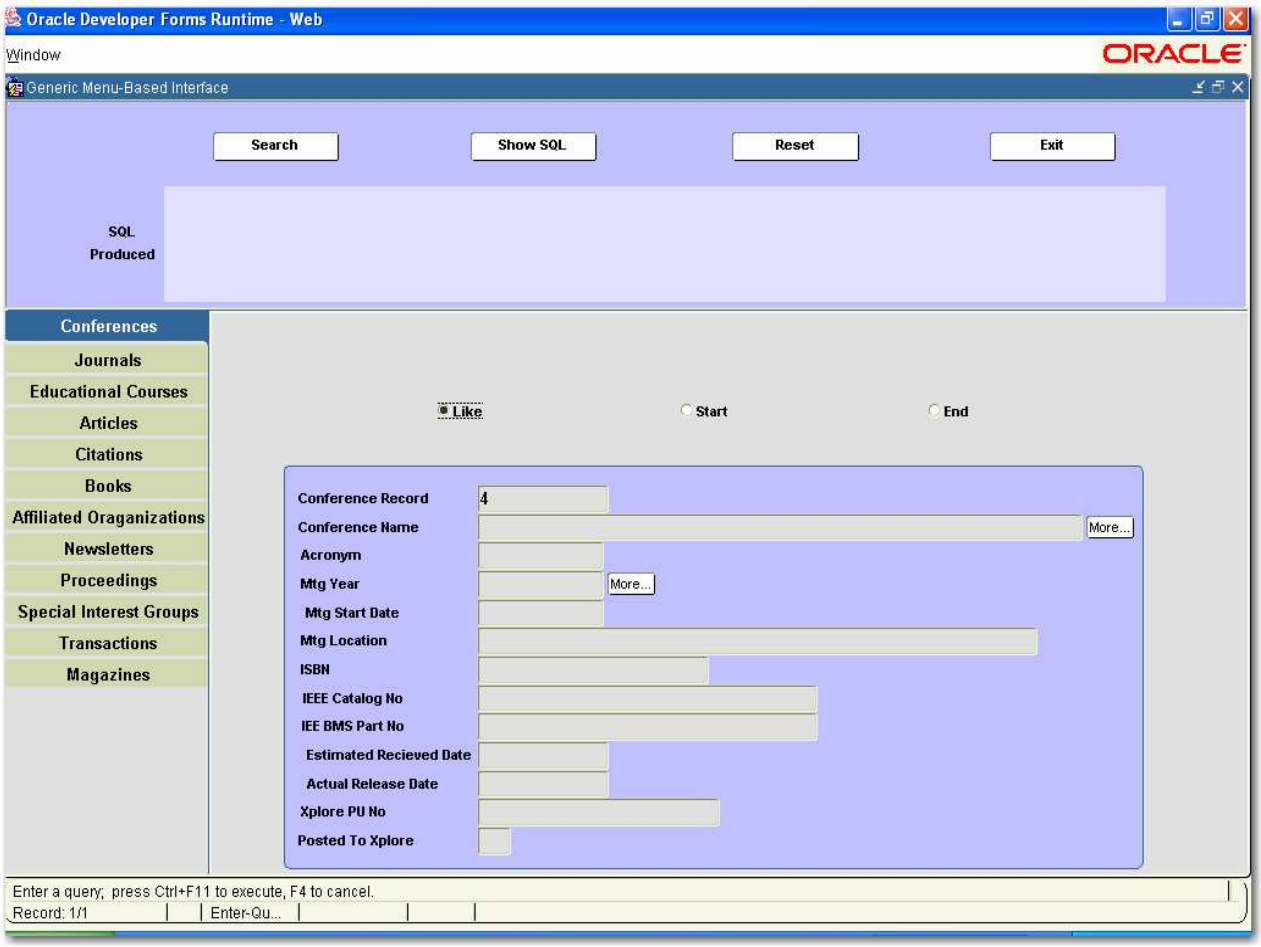


Figure 4.10: Searching for Conferences Depending on the Conference Record

Then, the searcher presses the 'Search' button to view the result of his search, and he could also press the 'Show SQL' button to view the SQL statement related to the search process. The search result and the records that satisfy the query is found and previewed easily and quickly to the user as shown in **Figure 4.11**. And the user could move from a record to another easily using the arrow buttons, and view all the record's information.

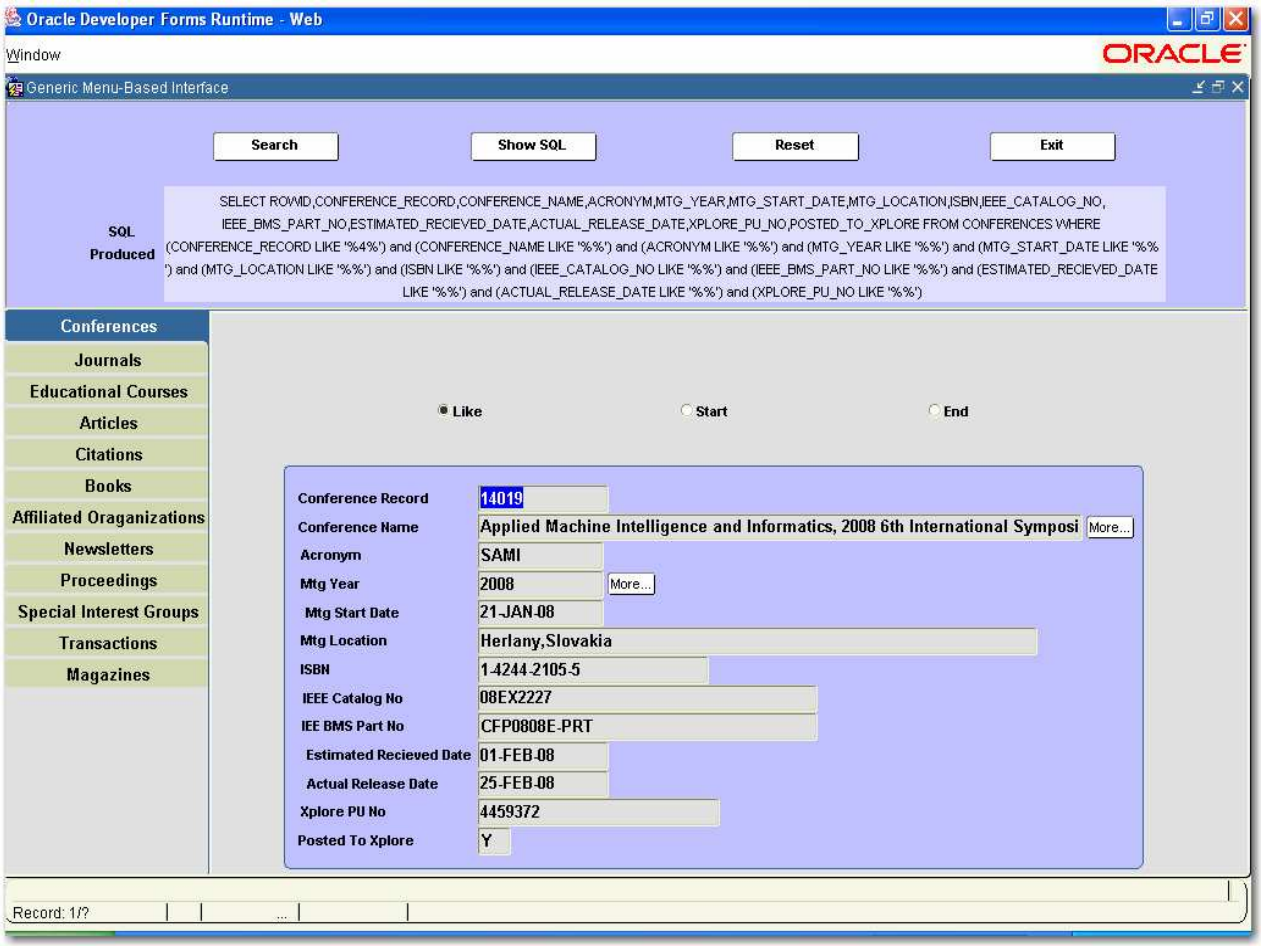


Figure 4.11: The Result of the Search Process Done in Example (2)

Example (3): This example will explain clearly the role of the control grammar structure used in building GMBI, and the great benefits of using the dynamic built-in menus during the search process. The user in this example searches for books in a specific area including main category, and sub category. First of all, the user accesses the 'Books' tab page. And then by pressing the 'More…' button related to the field 'Main Category', a pop-up menu will appear that includes all the records in the column Main_Category of table Books in the database. The user chooses the option 'Databases' as shown in **Figure 4.12**.

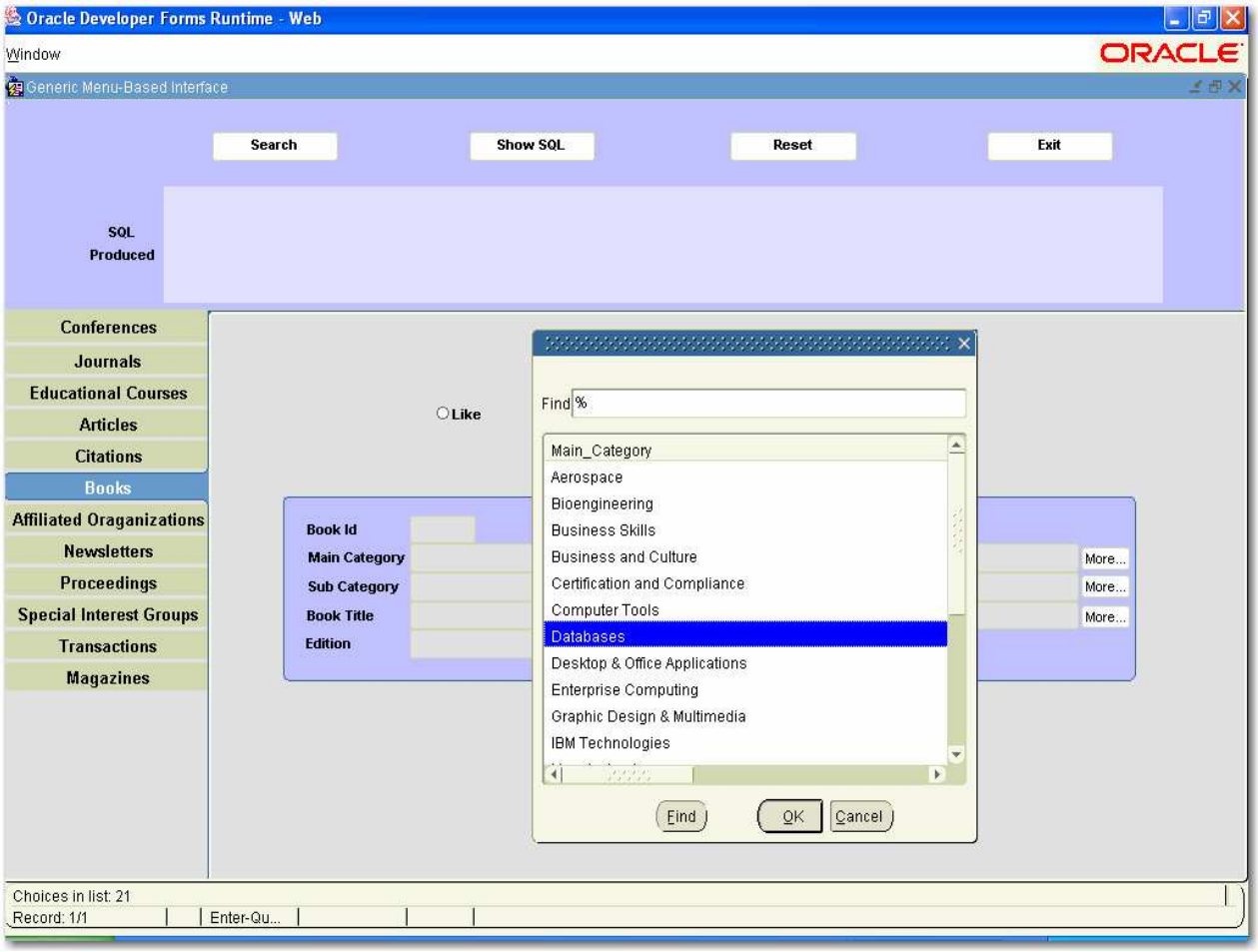


Figure 4.12: Searching for Books Using the Dynamic Built-In Menu ‘Main Category’

After selecting the main category, the user aims to select a specific sub category. Similarly, when he presses the ‘More…’ button related to the field ‘Sub Category’, a pop-up menu will appear that includes the records in the column Sub_Category of table Books in the database that only have the main category ‘Databases’ and match the query. In our example, the user chooses the option ‘Data Warehousing’ and then using the same way views the menu related to the field ‘Book Title’, which includes only the records in the column Book_Title of table Books that match the query and have the main category ‘Databases’ and the sub category ‘Data Warehousing’, and he chooses the title ‘Building the Data Warehouse’ as shown in **Figure 4.13**.

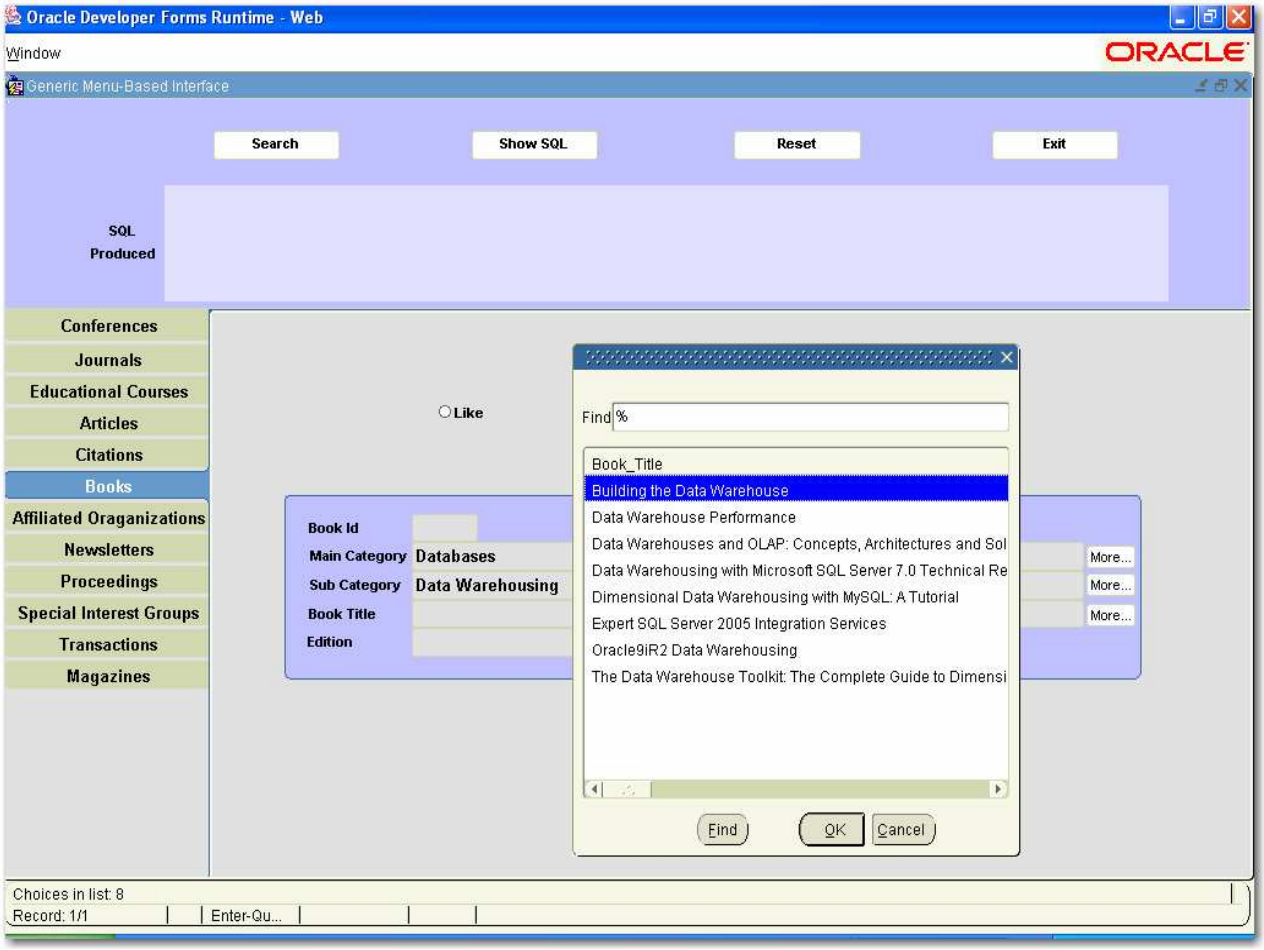


Figure 4.13: Continue Searching for Books Using the Dynamic Built-In Menu 'Book Title'

Then, the searcher presses the 'Search' button to view the result of his search, and he could also press the 'Show SQL' button to view the SQL statement related to the search process. Retrieving the search result, two records with two different Editions that satisfy the query is found and previewed, and the user could move between them easily using the arrow buttons. The search result of this example is shown in **Figure 4.14**.

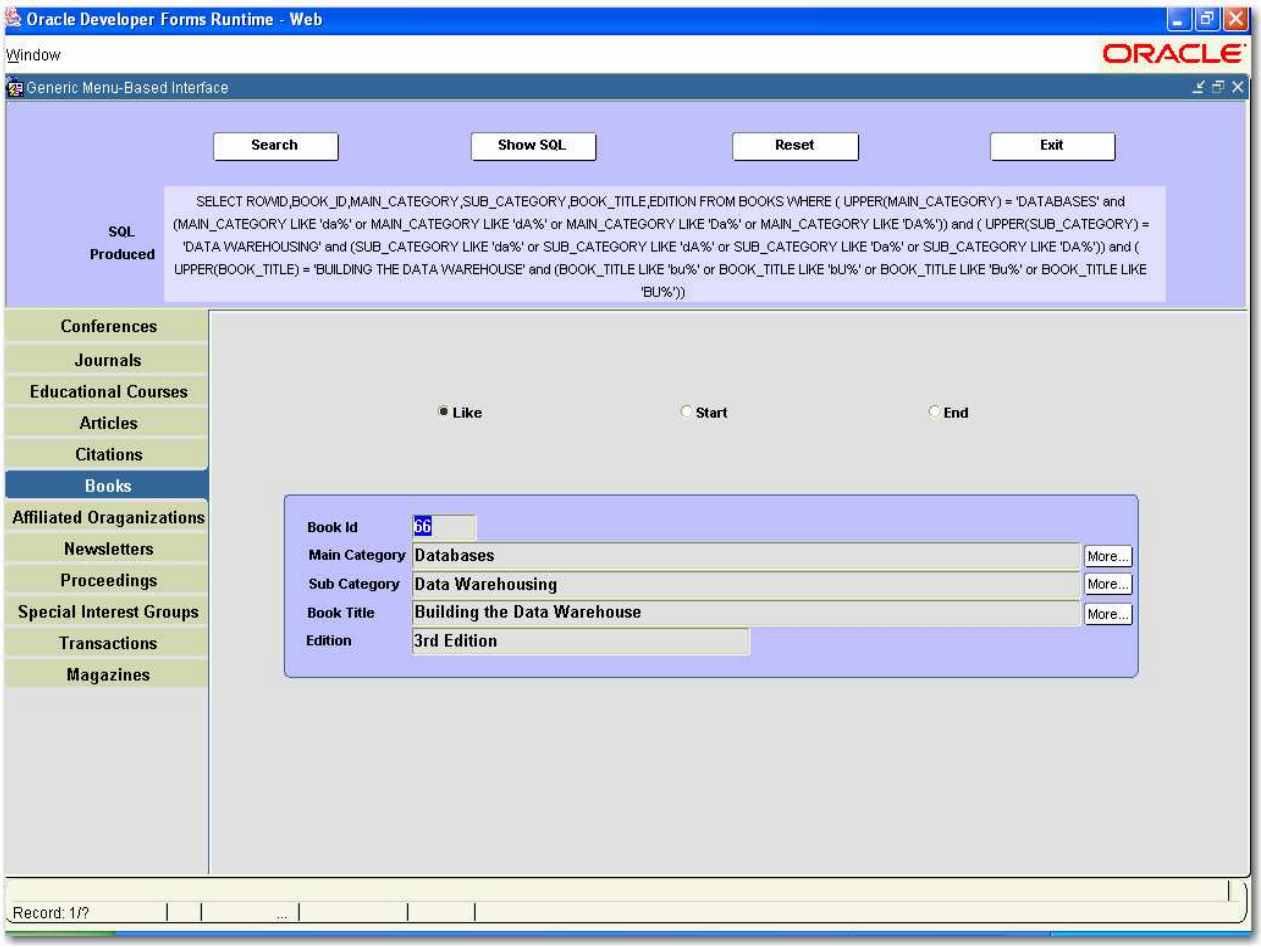


Figure 4.14: The Result of the Search Process Done in Example (3)

The previous examples present just a sample of the many features of GMBI. And by clarifying them, we claim that the results of the testing and evaluation phase and the many benefits of using GMBI are now clear and speak for themselves.

After the system is installed and put in practical use, maintenance involves correcting the errors which were not discovered in earlier stages of the development life cycle, improving the implementation of the system units and enhancing the system's services as new requirements are discovered.

Improving our Generic Menu-Based Interface components and enhancing the services it provides the users with, could always be in place to satisfy the users' needs. Evolving the Interface with its components and design would serve its functionality, and such a process could also be considered as future work for us and for other interested researchers.

CHAPTER 5

DISCUSSION AND RESULTS

In this chapter, we have presented and discussed the results of our work that were accomplished after completing the testing and validation phase. Then we are going to present the benefits and advantages that GMBI provides compared to other forms and search interfaces.

5.1 EXPERIMENTAL RESULTS ACCOMPLISHED AFTER TESTING AND VALIDATION PHASE

Users could use our GMBI to accomplish their search processes in several ways. They may enter a text into any field of any of the tab pages, and then press the 'Search' button and the search result will be displayed for them. They could also use the radio buttons mentioned earlier: 'Like', 'Start', and 'End' in parallel with typing the text in the field to improve their search.

Also, when they press the 'Show SQL' button during a search process, the SQL statement related to their search will be viewed for the users' benefit. This legitimate SQL statement is generated automatically, and is useful mostly for researchers and professional DB users. They could base on such query and produce more compound SQL statements which may give wider results, and may work successfully with different types of DBMS. This SQL statement is also important for application and integration purposes.

Furthermore, the most useful facility that GMBI offer for users during their search, is selecting a specific item from a menu that pop up when pressing one of the 'More…' buttons in the tab pages that is related to the item they are focusing their search on. And by pressing the 'Search' button, they can view the result. These menus are controlled by a control grammar structure that is if the user selects a specific item through some menu, then the other related menus in the same data block and tab page will display to the user only the items that go along with the first selected item to choose from, as was explained in **Example (3)** mentioned in section 4.4.

5.2 BENEFITS AND ADVANTAGES OF GMBI COMPARED TO OTHER SEARCH INTERFACES

In this section, we have viewed some search forms and interfaces, and illustrated the main differences between them and GMBI, and then introduced the benefits and advantages that GMBI presents.

Mendelzon et al. designed a Java applet invoked from an HTML form that is shown in **Figure 5.1**, in order to make WebSQL available to all WWW users. But most users are not familiar with formal query languages like WebSQL, and they probably face problems

in data retrieval because such interfaces burden them to learn, and recall precisely the query language and database structure [19].

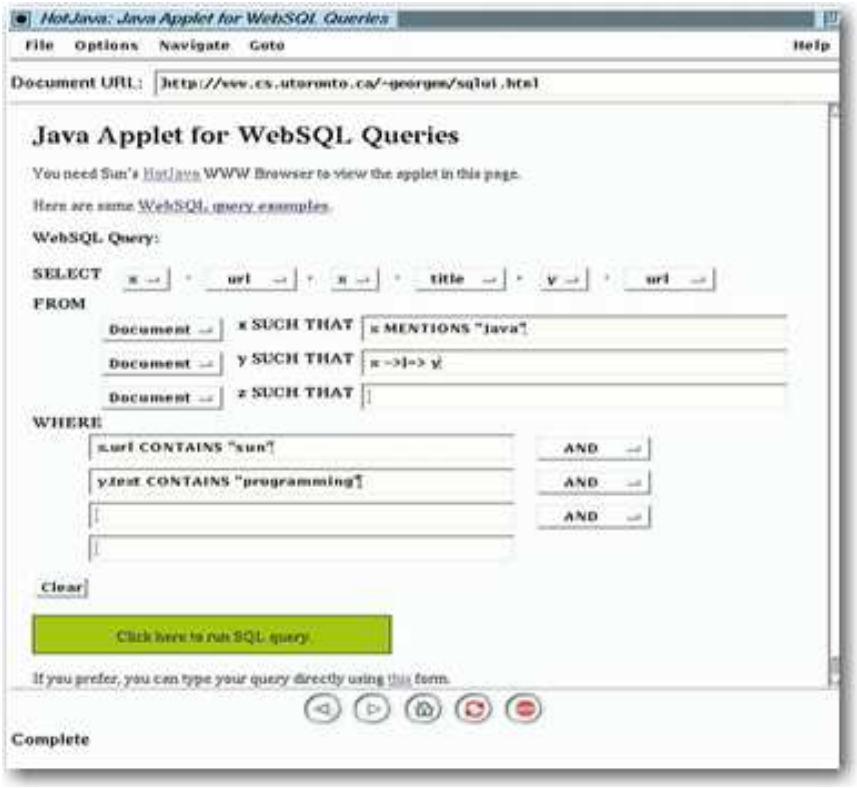


Figure 5.1: The WebSQL User-Interface [19]

Arocena et al. have developed three different interfaces. The simplest interface is an HTML form connected to a CGI script. The user can either fill in the form to assemble a query or type a complete WebSQL query directly. For Java-aware browsers, they have developed a much more user-controllable front-end under the form of Java applet which is shown in **Figure 5.2**. They also developed a stand-alone Java application that could be installed on the user's machine [1].

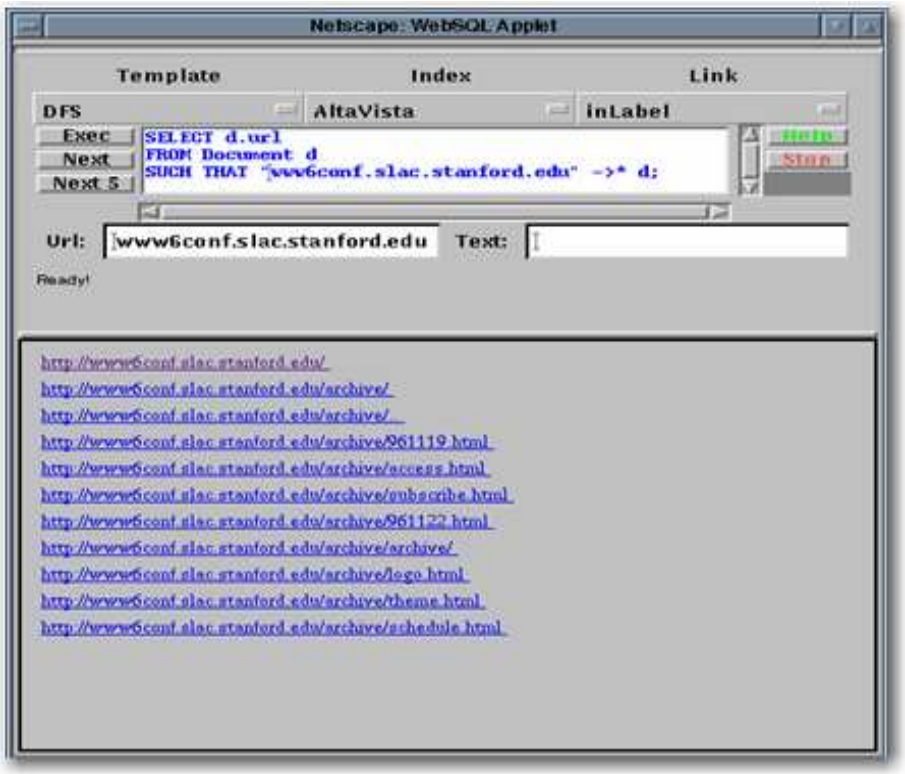


Figure 5.2: The WebSQL Applet [1]

The ACM SIGMOD Anthology is a digital library for the database systems research community developed by ACM SIGMOD, and it has developed the DBLP search. The DBLP server provides bibliographic information on major computer science journals and proceedings. Initially the server was focused on Database systems and Logic Programming (DBLP), now it is gradually being expanded toward other fields of computer science. So, "DBLP" could also be read as "Digital Bibliography & Library Project" [57]. The DBLP search form is shown in **Figure 5.3**. The user is required to fill the form and enter meaningful text in the search fields to be able to retrieve data.

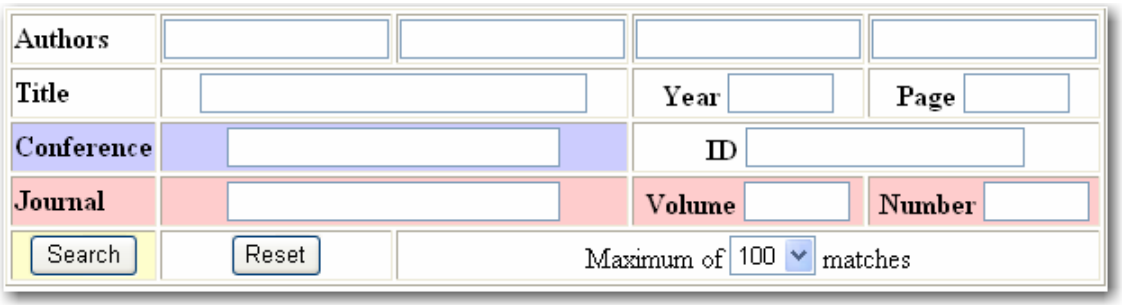


Figure 5.3: ACM SIGMOND Anthology DBLP Search [57]

Another search form is developed by the RFC (Request for Comments) series, which contains technical and organizational documents about the Internet, including the technical specifications and policy documents produced by the Internet Engineering Task Force (IETF) [54]. Their search form is shown in **Figure 5.4**. The user is also required to type what he is looking for in the search filed.

Figure 5.4: RFC Index Search Engine [54]

After overviewing the previous interfaces and search forms, and to make the picture more clear for the reader, presenting a comparison with our Generic Menu-Based Interface (GMBI) is a must. We are going to describe the features and characteristics that the GMID provides to the user and how it distinguishes from other Interfaces. **Table 5.1** presents a comparison between GMBI and the previous forms and interfaces.

Table 5.1: Comparison between GMBI and other search interfaces

Interface / Comparison	*Mendelzon*	*Arecona*	*DBLP*	*RFC*	*GMBI*
Menu-Based (Use Dynamic Menus)	**No**	**No**	**No**	**No**	**Yes**
Display the SQL St. Automatically	**No**	**No**	**No**	**No**	**Yes**
Generic to almost All Publications	**No**	**No**	**No**	**No**	**Yes**
Use Radio Buttons	**No**	**No**	**No**	**Yes**	**Yes**
Require Users to Type Text	**Yes**	**Yes**	**Yes**	**Yes**	**Not Required**
Require Users to Type SQL St.	**Yes**	**Yes**	**Not Required**	**Not Required**	**Not Required**

The main advantage that GMBI provides is using dynamic built-in menus while attending a search process, not acquiring the user to type a text or memorize it. Nevertheless, the user also has the ability to type any text or part of it and then uses the useful radio buttons to simplify his search. Furthermore, GMBI provides support and guidance on query formulation driven by a grammar-based control structure. The interface generates legitimate query step by step as menu choices and makes the users' mission easier.

Also, GMBI provides the user with the SQL statement that could work successfully with different types of DBMS, such as: Microsoft Access, Microsoft SQL Server, Oracle, My SQL, Sybase, and IBM DB2. This supports users in remembering the query syntax and metadata, which does not require data processing and database structure knowledge. The users also do not need to be familiar with formal query languages like SQL or WebSQL.

Furthermore, our interface also provides the users with the search result they are seeking successfully, and retrieve the needed data from the database. This will simplify the job of the Internet users and make their work more efficient in information retrieval.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

In this chapter, we have presented and discussed the conclusions of our work, and viewed our final results. Also, ideas for future work have been suggested at the end of the chapter.

6.1 SUMMARY

The WWW is a large, heterogeneous, distributed collection of documents connected by hypertext links. Searching the Internet and the WWW is a daily need to most people. Our thesis main goal was to develop a Generic Menu-Based Interface (GMBI) that supports users in information retrieval and through their search processes.

First of all, we have introduced an overview of Web querying and clarified the statement of problem, our goals and contribution, the previous related work and the software used during our work in chapter 1. In chapter 2 we gave an overview of various mapping and conversion tools and methodologies that convert and import Web data to different types of databases. Moreover, we described the process we used of converting Web data to relational database in details in chapter 3. We presented our domain of interest which is Web sites that focus on publications. Also, we described how Web data was converted to Microsoft Excel and then to Oracle database, using 'Toad for Oracle' tool. In chapter 4, we presented the phases of developing our Generic Menu-Based Interface (GMBI). We mainly described its design, implementation and testing. Furthermore, we presented our interface advantages and benefits, compared to other search forms and interfaces.

6.2 CONCLUSIONS

As the Web and its usage continues to grow, so grows the opportunity to analyze Web data and extract all manner of useful knowledge from it. Millions of users access the Web daily searching for various types of useful information. As those users differ in their proficiency and background knowledge, they face problems in having the right information from the right interface. Earlier, they had to be aware of various index servers that were deployed on the Web, of their strengths and weaknesses, and of the peculiarities of their query interfaces. As a matter of fact these queries cannot exploit the structure and topology of the document network. Also, most database interfaces provide poor guidance to Web users in query formulation, and require the users to be familiar with formal query languages like SQL, although most of those users are not used to such languages.

In our thesis, we have developed a Generic Menu-Based Interface (GMBI). The whole work consists of two main phases: phase one presents extracting the data from the Web and then converting it to a relational database, and phase two presents the analysis, design, implementation and testing phases of developing GMBI.

First of all, we analyzed different Web sites that concern publications, and made data and knowledge extraction from them in several levels. We also studied several mapping and conversion tools and tried them to make the data extraction from the Web and then convert this data to relational databases, but the database we have got and its data weren't valuable as the data we were seeking to get from the Web sites we have analyzed and studied. So that, we continued our search and found out other methodologies that serve our purposes in a better way.

We extracted the data needed from the Web sites using an Excel Extractor. Then, we modified the extracted data into well-formatted tables. Then our focus was to convert this data and import it into an oracle database using the tool 'Toad for Oracle', to use this database in the testing phase of developing our GMBI.

For those users who are not familiar with formal query languages and others, we have designed and successfully implemented a Generic Menu-Based Interface (GMBI) that will provide support and guidance on query formulation driven by a grammar-based control structure.

The main advantage that GMBI provides is using dynamic built-in menus while attending a search process, not acquiring the user to type a text or memorize it. The user could easily use the buttons and the menus to select the data he is searching for. Nevertheless, the user also has the ability to type any text or part of it and then uses the buttons to simplify his search.

After the testing and evaluation phase of our front-end interface, we proved that the interface generates legitimate query step by step as menu choices, and provide the user with the SQL statement that could work successfully with different types of DBMSs, such as: Microsoft Access, Microsoft SQL Server, Oracle, My SQL, Sybase, and IBM DB2. This supports users in remembering the query syntax and metadata, which does not require data processing and database structure knowledge.

Furthermore and most importantly, we proved that our interface also could provide users with the search result they are seeking successfully, and retrieve the needed data from the database. As a result, our work will simplify the job of the Internet users and make their work more efficient in information retrieval.

6.3 FUTURE WORK

We would like to suggest some interesting issues and ideas that could not be reached because of time, resources and other constraints and they will aid as an improvement on our GMBI. As future work, we mention:

- Applying an experimental test by allowing a group of searchers and Internet users to try out GMBI besides completing our testing phases, and assess the features and the benefits of the interface.

- Adding improvements and extra features to GMBI, and expanding the database to include other domains other than publications, such as: Sports, CD's, Music, etc.

- Developing other mapping and conversion tools that could directly map Web data to relational databases with full and valuable data.

- Proposing a solution for the problem that converting XML data don't give us the right and valuable data that we are seeking from Web sites.

References

[1] Arocena G. O., Mendelzon A. O. and Mihaila G. A., "Applications of a Web Query Language', **1997**. http://www.cs.toronto.edu/~websql/

[2] Asonov D., "Querying Databases Privately: A New Approach to Private Information Retrieval" (Lecture Notes in Computer Science), 1st edition, Springer, **2004**.

[3] Atay M., Chebotko A., Liu D., Lu S. and Fotouhi F., "Efficient schema-based XML-to-Relational data mapping", Department of Computer Science, Wayne State University, Detroit, MI 48202, USA, December **2005**.

[4] Borges J. and Levene M., "Mining association rules in hypertext databases". *In Proceeding of the fourth International Conference on Knowledge Discovery and Data Mining (KDD)*, New York , USA, August 27-31, **1998**.

[5] Bourret R., "XML and Databases", Septemper **2005**.

[6] Bourret R., "Mapping DTDs to Databases", May **2001**. http://www.xml.com/pub/a/2001/05/09/dtdtodbs.html

[7] Cafarella M. J., Re' C., Suciu D., Etzioni O. and Banko M., "Structured Querying of Web Text, A Technical Challenge", University of Washington, *3rd Biennial Conference on Innovative Data Systems Research (CIDR*), Asilomar, California, USA, January **2007**.

[8] Etzioni O., Cafarella M., Downey D., Kok S., Popescu A-M., Shaked T., Soderland S., Weld D. and Yates A., "Web-scale Information Extraction in KnowItAll", New York, May **2004**.

[9] Etzioni O., Cafarella M., Downey D., Popescu A-M., Shaked T., Soderland S., Weld D. and Yates A., "Methods for Domain-Independent Information Extraction from the Web: An Experimental Comparison", AAAI, San Jose, CA, July **2004**.

[10] Gajos K., Czerwinski M., Tan D. and Weld D., "Exploring the Design Space for Adaptive Graphical User Interfaces", Advanced Visual Interfaces (AVI), Venice, Italy, May **2006**.

[11] Gracia J., Trillo R., Espinoza M. and Mena E., "Querying the Web: A Multiontology Disambiguation Method", University of Zaragoza, Spain, July **2006**.

[12] James Bailey J., Bry F., Furche T. and Schaffert S., " Web and Semantic Web Query Languages: A Survey", The University of Melbourne, Victoria, Australia, and University of Munich, M¨unchen, Germany, **2005**.

[13] Kaufmann E., Bernstein A. and Zumstein R., Querix: "A Natural Language Interface to Query Ontologies Based on Clarification Dialogs", University of Zurich. *In proceedings of the 5th International Semantic Web Conference (ISWC)*, Athens, GA, November **2006**.

[14] Konopnicki D. and Shmueli O., "W3QL: A query system for the World Wide Web". In *Proceedings of the 21st International Conference on Very Large Databases*, Zurich, Switzerland, **1995**.

[15] Kosala R., "Web Mining Research: A Survey", Katholieke Universiteit Leuven, Celestijnenlaan 200A, B3001 Heverlee, Belgium, ACM SIGKDD, July **2000**.

[16] Lacroix Z., Sahuguet A. and Chandrasekar R., "Information Extraction and Database Techniques: A user-oriented approach to querying the Web", University of Pennsylvania, **1998**.

[17] Lakshmanan L. V. S., Sadri F., and Subramanian I. N., "'A declarative language for querying and restructuring the Web". *In: Proc. 6th. International Workshop on Research Issues in Data Engineering (RIDE)*, New Orleans, February **1996**.

[18] Mendelzon A. O., Mihaila G. A. and Milo T., "Querying the World Wide Web". In *Proc. of First Int. Conf. on Parallel and Distributed Information Systems (PDIS)*, pp. 80-91, December **1996**. ftp://db.toronto.edu/pub/papers/pdis96.ps.gz

[19] Mendelzon A. O. and Milo T., "Formal models of Web queries". *In: Proc. Acm Pods* **1997**.

[20] Mihaila G. A., "WebSQL - an SQL-like query language for the World Wide Web". Master's thesis, University of Toronto, **1996**.

[21] Mihaila G. A., WebSQL Home Page, **1997**.
http://www.cs.toronto.edu/~georgem/WebSQL.html

[22] Minohara T. and Watanabe R., "Queries on structure in hypertext". *In: Foundations of Data Organization and Algorithms (FODO)*, Berlin Heidelberg New York: Springer, **1993**.

[23] Morville P., "Information Architecture on the World Wide Web", pp:58, February **1998**.

[24] Naughton J.et al., "The Niagara Internet Query System", Computer Sciences Department, University of Wisconsin-Madison, **2000**.

[25] Oracle Inc., Statement of Direction, "Oracle Forms – Oracle Reports – Oracle Designer", November **2007**.

[26] Patel B. C. and Sunderraman R., "Querying Web Data: An Object-Oriented Approach"**,** *ACM Southeast Regional Conference, Proceedings of the 38th annual on Southeast Regional Conference,* Georgia State University, **2000**.

[27] Petropoulos M., Papakonstantinou Y. and Vassalos V., "Building XML Query Forms and Reports with XQForms", **2003**.

[28] Piotrowski R., "Natural language interface to legal databases", Umea University, **2005**.

[29] Qutaihsat M. A., "An Arabic Menu-Based Natural Language Interface to Database Systems", *Proceedings of 8th International Conference on Systems Research Informatics and Cybernetics*, Germany, **1995**.

[30] Rajaraman A., "Virtual Database Technology, XML, and the Evolution of the Web". *Position paper, Junglee Corporation*, **1998**. Available at http://www.junglee.com/tech/xml_pos.html

[31] Ramakrishnan R. and Gehrke J., "Database Management Systems", Third edition, August **2002**.

[32] Smith D. and Lopez M., "Information extraction for semi-structured documents". In *Proceedings of the Workshop on Management of Semistructured Data, In conjunction with PODS/SIGMOD*, Ventana Canyon Resort, Tucson, Arizona, **1997**.

[33] Soderland S., Etzioni O., Shaked T. and Weld D., "The Use of Web-based Statistics to Validate Information Extraction," *(to appear) AAAI workshop on Adaptive Text Extraction and Mining*, San Jose, CA, July **2004**.

[34] Sommerville I., "Software Engineering", Seventh Edition, Pearson Addison Wesely, **2007**.

[35] Srivastava J., Desikan P. and Kumar V., "Web Mining – Concepts, Applications and Research Directions", University of Minnesota, *AHPCRC Technical Report*, **2005**.

[36] Sunderraman R., ReQueSS, "Relational Querying of Semi-Structured Data". *International Conference on Data Engineering (ICDE), Demonstration Paper*, San Diego, CA, February **2000**.

[37] Taniar D., "Web Information Systems", Idea Group Pub, **2004**.

[38] Vertan C., "Querying Multilingual Semantic Web in Natural Language", University of Hamburg, **2004**.

[39] Weld D., Anderson C., Domingos P., Etzioni O., Lau T., Gajos K. and Wolfman S., "Automatically Personalizing User Interfaces" *(IJCAI)*, August **2003**.

[40] Yates A., Etzioni O. and Weld D., "A Reliable Natural Language Interface to Household Appliances" *(IUI)*, January **2003**.

[41] Yates R. B. and Neto B. R., "Modern Information Retrieval". Addison-Wesely Longman Publishing Company, **1999**.

WEB RESOURCES

[42] http://www.acme.com/searches.html
Access date: May 4, 2008

[43] http://www.altova.com/products/mapforce/data_mapping.html
Access date: April 24, 2008

[44] http://www.altova.com/products/stylevision/xslt_stylesheet_designer.html
Access date: April 24, 2008

[45] http://www.altova.com/products/xmlspy/xml_editor.html
Access date: April 23, 2008

[46] http://www.delphion.com
Access date: May 4, 2008

[47] http://www.en.wikipedia.org/wiki/Site_map
Access date: April 22, 2008

[48] https://www.google.com/webmasters/tools/docs/en/protocol.html
Access date: April 21, 2008

[49] http://www.hitsw.com/index.html
Access date: April 25, 2008

[50] http://www.lib.uchicago.edu/efts/ARTFL/projects/encyc/searchform.html
Access date: May 5, 2008

[51] http://www.oracle.com/technology/products/
Access date: April 17, 2008

[52] http://www.oxygenxml.com/
Access date: April 23, 2008

[53] http://www.quest.com/toad/
Access date: April 17, 2008

[54] http://www.rfc-editor.org/rfcsearch.html
Access date: May 5, 2008

[55] http://www.rocketface.com/maintain_website/ror_rss_sitemap_generator.html
Access date: April 22, 2008

[56] http://www.scriptol.com/seo/simple-map.html
Access date: April 22, 2008

[57] http://www.sigmod.org/sigmod/anthology/index.htm
Access date: May 6, 2008

[58] http://www.sitemaps.org/
Access date: April 21, 2008

[59] http://www.sitemapspal.com/
Access date: April 21, 2008

[60] http://www.stylusstudio.com/xml_product_index.html
Access date: April 26, 2008

[61] http://www.w3.org/XML/Schema
Access date: April 22, 2008

[62] http://www.winsite.com/
Access date: April 23, 2008

[63] http://www.xml-sitemaps.com/
Access date: April 21, 2008

[64] http://www.xmlwrench.com/
Access date: April 23, 2008

[65] http://www.math.tau.ac.il/~matias/courses/sem_fall98/websql/wwwSQL.htm
Written by Golan Weiss, Access date: May 17, 2008

APPENDICIES

APPENDIX A
Glossary of Acronyms

ACM	Association for Computing Machinery.
ADF	Application Development Framework.
AMID	Arabic Menu-Based Natural Language Interface to Database Systems.
CGI	Common Gateway Interface.
CSS	Cascading Style Sheets.
DBA	Database Administrator.
DBMS	Database Management Systems.
DDL	Data Definition Language.
DOM	Document Object Model.
DTDs	Document Type Definitions.
EDI	Electronic Data Interchange.
ExDB	Extraction Database.
FTP	File Transfer Protocol.
GMBI	Generic Menu-Based Interface.
HTB	Horizontal Toolbar.
HTTP	Hypertext Transfer Protocol.
IBM	International Business Machines Corporation.
IE	Information Extraction.
IEEE	Institute of Electrical and Electronics Engineers.
J2EE	Java Platform, Enterprise Edition.
LOVs	Lists of Values.
Mtg	Meeting.
ODBC	Object Database Connection.
PHP	Hypertext Preprocessor.
PL/SQL	Procedural Language extensions to Structured Query Language.
RDB	Relational Database.

RDBMS	Relational Database Management System.
ROR	Resources of a Resource.
RSS	Really Simple Syndication.
SAX	The Simple Application programming interface for The Extensible Markup Language.
SDLC	Systems Development Lifecycle.
SOA	Service-Oriented Architectures.
SOAP	Service-Oriented Architecture Protocol.
SQL	Structured Query Language.
UML	Unified Modeling Language.
V & V	Verification and Validation.
VTB	Vertical Toolbar.
W3QL	World-Wide Web Query Language.
WSDL	Web Services Description Language.
WWW	World Wide Web.
XHTML	Extensible Hypertext Markup Language.
XML-QL	Extensible Markup Language-Query Language.
XSL	eXtensible Stylesheet Language.
XSLT	eXtensible Stylesheet Language Transformations.

APPENDIX B
The Software Life Cycle

The Systems Development Lifecycle (SDLC) process is defined as an organized way to determine customer needs and user requirements such that technology can be applied through systems development, and help customers and users perform their jobs more effectively and efficiently [34].

The SDLC process ends with maintenance and sustainment activities but includes a way to use feedback for continuous improvement of processes and systems. Project management is a tool used to manage the use of a systems development methodology (a structured approach to systems development), and ensure systems are built that help the users and customers [34].

The traditional Waterfall model, is a sequential software development model and a process for the creation of software in which development is seen as flowing steadily downwards (like a waterfall) through the development phases as shown in **Figure B.1** [34].

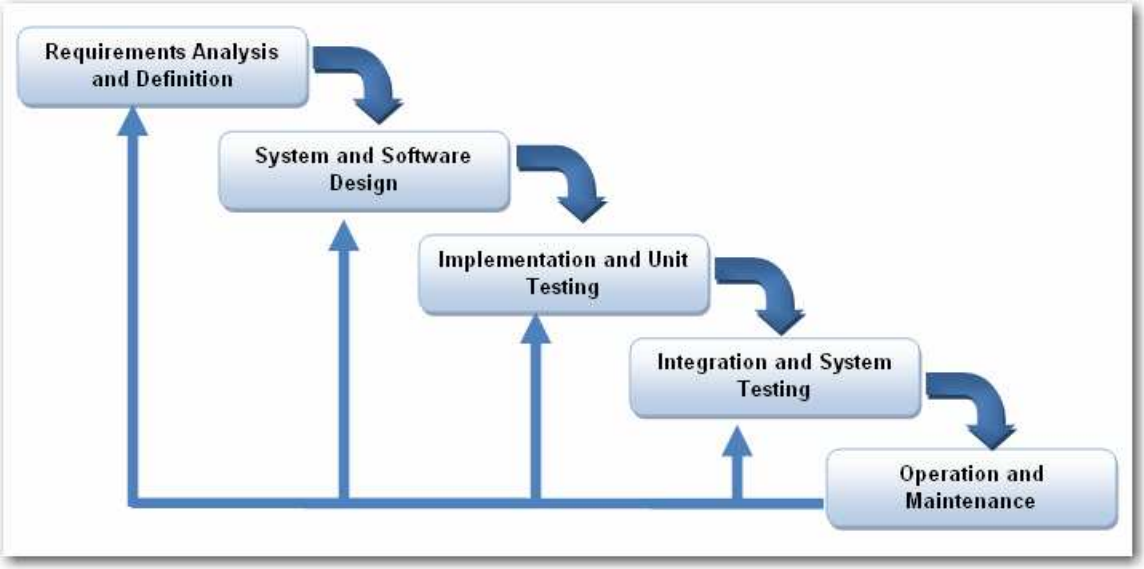


Figure B.1: The Software Life Cycle (Waterfall Model)

The principal stages of the model map onto fundamental development activities:

1) *Requirements analysis and definition:* The System's services, constraints and goals are established by consultation with system users. They are then defied in detail and serve as a system specification.

2) *System and software design:* The systems design process partitions the requirements to either hardware or software systems. It establishes an overall

system architecture. Software design involves identifying and describing the fundamental software system abstractions and their relationships.

3) *Implementation and unit testing:* During this stage, the software design is realized as a set of programs or program units. Unit testing involves verifying that each unit meets its specification.

4) *Integration and system testing:* The individual program units or programs are integrated and tested as a complete system to ensure that the software requirements have been met. After testing, the software system is delivered to the customer.

5) *Operation and maintenance:* Normally (although not necessarily) this is the longest life-cycle phase. After the system is installed and put in practical use, maintenance involves correcting the errors which were not discovered in earlier stages of the development life cycle, improving the implementation of the system units and enhancing the system's services as new requirements are discovered.

In principle, the result of each phase is one or more documents that are approved ('signed off'). The following phase should not start until the previous phase has finished. In practice, these stages overlap and feed information to each other. During design, problems with requirements are identified; during coding, design problems are found and so on. The software process is not a simple linear model but involves a sequence of iterations of the development activities.

Software validation or, more generally, verification and validation (V & V) is intended to show that a system conforms to its specification and that the system meets the expectations of the users. Systems should not be tested as a single, monolithic unit. **Figure B.2** shows a three-stage testing process where system components are tested, the integrated system is tested and, finally, the system is tested with real data [34].

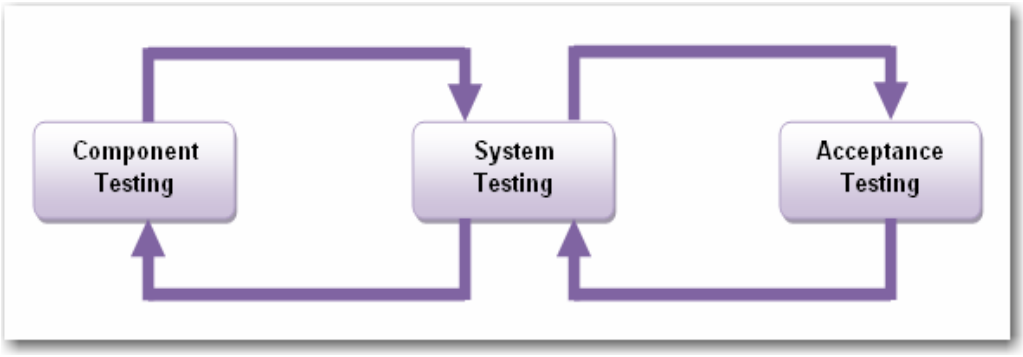


Figure B.2: Testing Process Stages

The stages in the testing process are:

1) *Component (or unit) testing:* Individual components are tested to ensure that they operate correctly. Each component is tested independently, without other system components. Components may be simple entities as functions, object classes or triggers, or may be coherent groupings of these entities.

2) *System Testing:* The components are integrated to make up the system. This process is concerned with finding errors that result from unanticipated interactions between components and component interface problems. It is also concerned with validating that the system meets its requirements and testing the emergent system properties.

3) *Acceptance testing:* This is the final stage in the testing process before the system is accepted for operational use. The system is tested with real data rather than with simulated test data.

APPENDIX C
The Code of Developing GMBI

- Triggers on form level:

- **PRE-FORM:**

```
set_window_property (forms_mdi_window,window_state,maximize);
set_window_property ('window1',window_state,maximize);
```

- **WHEN-TAB-PAGE-CHANGED:**

```
if :system.tab_new_page='CONFERENCES' then
    :global.tab:=1;
elsif :system.tab_new_page='JOURNALS' then
    :global.tab:=2;
elsif :system.tab_new_page='EDUCATIONAL_COURSES' then
    :global.tab:=3;
elsif :system.tab_new_page='ARTICLES' then
    :global.tab:=4;
elsif :system.tab_new_page='CITATIONS' then
    :global.tab:=5;
elsif :system.tab_new_page='BOOKS' then
    :global.tab:=6;
elsif :system.tab_new_page='AFFILIATED_ORGANIZATIONS' then
    :global.tab:=7;
elsif :system.tab_new_page='NEWSLETTERS' then
    :global.tab:=8;
elsif :system.tab_new_page='PROCEEDINGS' then
    :global.tab:=9;
elsif :system.tab_new_page='SPECIAL_INTEREST_GROUPS' then
    :global.tab:=10;
elsif :system.tab_new_page='TRANSACTIONS' then
    :global.tab:=11;
```

```
elsif :system.tab_new_page='MAGAZINES' then
    :global.tab:=12;
end if;

if :system.mode='ENTER-QUERY' then
    exit_form;
end if;

:SQL:=NULL;
go_block(:system.tab_new_page);
enter_query;
```

- **ON-MESSAGE:**

```
declare
    v varchar2(1000);
begin
    v:=:system.tab_new_page;
if message_code=40353 then
    null;
elsif message_code=40301 then
    if show_alert('qry')=88 then
        raise form_trigger_failure;
    end if;
else
    message(message_type||'-'||message_code||': '||message_text);
end if;
end;
```

- **WHEN-NEW-FORM-INSTANCE:**

```
:global.tab:=1;
```

- Triggers on data block level:

- **WHEN-NEW-BLOCK-INSTANCE:**

```
enter_query;
```

- Triggers on item level:

- **WHEN_BUTTON_PRESSED on Push Button “MTG_YR”:**

```
if show_lov('CONF_MTG_YEAR_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “CONF_NAME”:**

```
if show_lov('CONF_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “JOUR_N”:**

```
if show_lov('JOUR_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “EDU_CRS_N”:**

```
if show_lov('EDU_COURS_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “ART_AUTHOR”:**

```
if show_lov('ART_AUTH_LOV')=true then
    set_item_property('articles.tit',visible,property_true);
    set_item_property('articles.tit',enabled,property_true);
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “TITLE”:**

```
if show_lov('ART_TIT_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “ARTICLE_YR”:**

```
if show_lov('ART_YR_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “CITATION”:**

```
if show_lov('CIT_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “CIT_AUTHOR”:**

```
if show_lov('CIT_AUTH_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “CIT_YR”:**

```
if show_lov('CIT_YR_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “MAIN_CATEGORY”:**

```
if show_lov('BOOKS_MAIN_LOV')=true then
    set_item_property('BOOKS.s',visible,property_true);
    set_item_property('BOOKS.s',enabled,property_true);
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “SUB_CATEGORY”:**

```
if show_lov('BOOKS_sub_LOV')=true then
    set_item_property('BOOKS.b',visible,property_true);
    set_item_property('BOOKS.b',enabled,property_true);
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “BOOK_TITLE”:**

```
if show_lov('BOOKS_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “AFF_ORG”:**

```
if show_lov('AFF_ORG_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “AFF_PUB”:**

```
if show_lov('AFF_PUB_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “NEWSLETTER”:**

```
if show_lov('NEW_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “PROCEEDING”:**

```
if show_lov('PROC_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “SIG”:**

```
if show_lov('SIG_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “TRANSACTION”:**

```
if show_lov('TRAN_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “MAGAZINE”:**

```
if show_lov('MAG_LOV')=true then
    null;
end if;
```

- **WHEN_BUTTON_PRESSED on Push Button “SEARCH”:**

```
if :global.tab=1 then

        if :conferences.c=2 then
            :CONFERENCE_RECORD:='%'||:CONFERENCE_RECORD||'%';
            :CONFERENCE_NAME:='%'||:CONFERENCE_NAME||'%';
            :ACRONYM:='%'||:ACRONYm||'%';
            :MTG_YEAR:='%'||:MTG_YEAR||'%';
            :MTG_START_DATE:='%'||:MTG_START_DATE||'%';
            :MTG_LOCATION:='%'||:MTG_LOCATION||'%';
            :ISBN :='%'|| :ISBN||'%';
            :IEEE_CATALOG_NO:='%'||  :IEEE_CATALOG_NO||'%';
            :IEEE_BMS_PART_NO:='%'|| :IEEE_BMS_PART_NO||'%';
            :ESTIMATED_RECIEVED_DATE:='%'|| :ESTIMATED_RECIEVED_DATE||'%';
            :ACTUAL_RELEASE_DATE :='%'||:ACTUAL_RELEASE_DATE||'%';
            :XPLORE_PU_NO :='%'||:XPLORE_PU_NO||'%';
```

```
elsif :conferences.c=3 then
    :CONFERENCE_RECORD:=:CONFERENCE_RECORD||'%';
    :CONFERENCE_name:=:CONFERENCE_name||'%';
    :ACRONYM:=:ACRONYm||'%';
    :MTG_YEAR:=:MTG_YEAR||'%';
    :MTG_START_DATE:=:MTG_START_DATE||'%';
    :MTG_LOCATION:=:MTG_LOCATION||'%';
    :ISBN :=      :ISBN||'%';
    :IEEE_CATALOG_NO:=:IEEE_CATALOG_NO||'%';
    :IEEE_BMS_PART_NO:=:IEEE_BMS_PART_NO||'%';
    :ESTIMATED_RECIEVED_DATE:= :ESTIMATED_RECIEVED_DATE||'%';
    :ACTUAL_RELEASE_DATE :=:ACTUAL_RELEASE_DATE||'%';
    :XPLORE_PU_NO :=:XPLORE_PU_NO||'%';

elsif :conferences.c=4 then
    :CONFERENCE_RECORD:='%'||:CONFERENCE_RECORD;
    :CONFERENCE_name:='%'||:CONFERENCE_name;
    :ACRONYM:='%'||:ACRONYm;
    :MTG_YEAR:='%'||:MTG_YEAR;
    :MTG_START_DATE:='%'||:MTG_START_DATE;
    :MTG_LOCATION:='%'||:MTG_LOCATION;
    :ISBN :='%'|| :ISBN||'%';
    :IEEE_CATALOG_NO:='%'||  :IEEE_CATALOG_NO;
    :IEEE_BMS_PART_NO:='%'|| :IEEE_BMS_PART_NO;
    :ESTIMATED_RECIEVED_DATE:='%'|| :ESTIMATED_RECIEVED_DATE;
    :ACTUAL_RELEASE_DATE :='%'||:ACTUAL_RELEASE_DATE;
    :XPLORE_PU_NO :='%'||:XPLORE_PU_NO;
end if;

elsif :global.tab=2 then
```

```
    if :journals.c=2 then
        :JOURNALS.JOURNAL_NO:='%'||:JOURNALS.JOURNAL_NO||'%';
        :JOURNALS.JOURNAL:='%'||:JOURNALS.JOURNAL||'%';
        :JOURNALS.ABBREVIATION:='%'||:JOURNALS.ABBREVIATION||'%';
        :JOURNALS.SOURCE:='%'||:JOURNALS.SOURCE||'%';

    elsif :journals.c=3 then
        :JOURNALS.JOURNAL_NO:=:JOURNALS.JOURNAL_NO||'%';
        :JOURNALS.JOURNAL_NO:=:JOURNALS.JOURNAL||'%';
        :JOURNALS.JOURNAL_NO:=:JOURNALS.ABBREVIATION||'%';
        :JOURNALS.JOURNAL_NO:=:JOURNALS.SOURCE||'%';

    elsif :journals.c=4 then
        :JOURNALS.JOURNAL_NO:='%'||:JOURNALS.JOURNAL_NO;
        :JOURNALS.JOURNAL:='%'||:JOURNALS.JOURNAL;
        :JOURNALS.ABBREVIATION:='%'||:JOURNALS.ABBREVIATION;
        :JOURNALS.SOURCE:='%'||:JOURNALS.SOURCE;
    end if;

elsif :global.tab=3 then

    if :EDUCATIONAL_COURSES.c=2 then
        :EDUCATIONAL_COURSES.COURSE_ID:='%'||:EDUCATIONAL_COURSES.C
        OURSE_ID||'%';
        :EDUCATIONAL_COURSES.COURSE_TITLE:='%'||:EDUCATIONAL_COURS
        ES.COURSE_TITLE||'%';
        :EDUCATIONAL_COURSES.COURSE_INSTRUCTORS:='%'||:EDUCATIONAL
        _COURSES.COURSE_INSTRUCTORS||'%';
        :EDUCATIONAL_COURSES.SPONSORED_BY:='%'||:EDUCATIONAL_COURS
        ES.SPONSORED_BY||'%';
        :EDUCATIONAL_COURSES.PUBLICATION_DATE:='%'||:EDUCATIONAL_C
        OURSES.PUBLICATION_DATE||'%';
```

```
:EDUCATIONAL_COURSES.COURSE_SUMMARY:='%'||:EDUCATIONAL_CO
URSES.COURSE_SUMMARY||'%';

elsif :EDUCATIONAL_COURSES.c=3 then
:EDUCATIONAL_COURSES.COURSE_ID:=:EDUCATIONAL_COURSES.COU
RSE_ID||'%';
:EDUCATIONAL_COURSES.COURSE_TITLE:=:EDUCATIONAL_COURSES.C
OURSE_TITLE||'%';
:EDUCATIONAL_COURSES.COURSE_INSTRUCTORS:=:EDUCATIONAL_CO
URSES.COURSE_INSTRUCTORS||'%';
:EDUCATIONAL_COURSES.SPONSORED_BY:=:EDUCATIONAL_COURSES.
SPONSORED_BY||'%';
:EDUCATIONAL_COURSES.PUBLICATION_DATE:=:EDUCATIONAL_COUR
SES.PUBLICATION_DATE||'%';
:EDUCATIONAL_COURSES.COURSE_SUMMARY:=:EDUCATIONAL_COUR
SES.COURSE_SUMMARY||'%';

elsif :EDUCATIONAL_COURSES.c=4 then
:EDUCATIONAL_COURSES.COURSE_ID:='%'||:EDUCATIONAL_COURSES.C
OURSE_ID;
:EDUCATIONAL_COURSES.COURSE_TITLE:='%'||:EDUCATIONAL_COURS
ES.COURSE_TITLE;
:EDUCATIONAL_COURSES.COURSE_INSTRUCTORS:='%'||:EDUCATIONAL
_COURSES.COURSE_INSTRUCTORS;
:EDUCATIONAL_COURSES.SPONSORED_BY:='%'||:EDUCATIONAL_COURS
ES.SPONSORED_BY;
:EDUCATIONAL_COURSES.PUBLICATION_DATE:='%'||:EDUCATIONAL_C
OURSES.PUBLICATION_DATE;
:EDUCATIONAL_COURSES.COURSE_SUMMARY:='%'||:EDUCATIONAL_CO
URSES.COURSE_SUMMARY;
end if;
```

```
elsif :global.tab=4 then

    if :ARTICLES.c=2 then
        :ARTICLES.ARTICLE_NO:='%'||:ARTICLES.ARTICLE_NO||'%';
        :ARTICLES.AUTHORS:='%'||:ARTICLES.AUTHORS||'%';
        :ARTICLES.ARTICLE_TITLE:='%'||:ARTICLES.ARTICLE_TITLE||'%';
        :ARTICLES.JOURNAL:='%'||:ARTICLES.JOURNAL||'%';
        :ARTICLES.ARTICLE_YEAR:='%'||:ARTICLES.ARTICLE_YEAR||'%';

    elsif :ARTICLES.c=3 then
        :ARTICLES.ARTICLE_NO:=:ARTICLES.ARTICLE_NO||'%';
        :ARTICLES.AUTHORS:=:ARTICLES.AUTHORS||'%';
        :ARTICLES.ARTICLE_TITLE:=:ARTICLES.ARTICLE_TITLE||'%';
        :ARTICLES.JOURNAL:=:ARTICLES.JOURNAL||'%';
        :ARTICLES.ARTICLE_YEAR:=:ARTICLES.ARTICLE_YEAR||'%';

    elsif :ARTICLES.c=4 then
        :ARTICLES.ARTICLE_NO:='%'||:ARTICLES.ARTICLE_NO;
        :ARTICLES.AUTHORS:='%'||:ARTICLES.AUTHORS;
        :ARTICLES.ARTICLE_TITLE:='%'||:ARTICLES.ARTICLE_TITLE;
        :ARTICLES.JOURNAL:='%'||:ARTICLES.JOURNAL;
        :ARTICLES.ARTICLE_YEAR:='%'||:ARTICLES.ARTICLE_YEAR;
    end if;

elsif :global.tab=5 then

    if :CITATIONS.c=2 then
        :CITATIONS.CITATION_NO:='%'||:CITATIONS.CITATION_NO||'%';
        :CITATIONS.AUTHORS:='%'||:CITATIONS.AUTHORS||'%';
        :CITATIONS.CITATION:='%'||:CITATIONS.CITATION||'%';
        :CITATIONS.JOURNAL:='%'||:CITATIONS.JOURNAL||'%';
        :CITATIONS.CITATION_YEAR:='%'||:CITATIONS.CITATION_YEAR||'%';
```

```
    elsif :CITATIONS.c=3 then
        :CITATIONS.CITATION_NO:=:CITATIONS.CITATION_NO||'%';
        :CITATIONS.AUTHORS:=:CITATIONS.AUTHORS||'%';
        :CITATIONS.CITATION:=:CITATIONS.CITATION||'%';
        :CITATIONS.JOURNAL:=:CITATIONS.JOURNAL||'%';
        :CITATIONS.CITATION_YEAR:=:CITATIONS.CITATION_YEAR||'%';

    elsif :CITATIONS.c=4 then
        :CITATIONS.CITATION_NO:='%'||:CITATIONS.CITATION_NO;
        :CITATIONS.AUTHORS:='%'||:CITATIONS.AUTHORS;
        :CITATIONS.CITATION:='%'||:CITATIONS.CITATION;
        :CITATIONS.JOURNAL:='%'||:CITATIONS.JOURNAL;
        :CITATIONS.CITATION_YEAR:='%'||:CITATIONS.CITATION_YEAR;
    end if;

elsif :global.tab=6 then

    if :BOOKS.c=2 then
        :BOOKS.BOOK_ID:='%'||:BOOKS.BOOK_ID||'%';
        :BOOKS.MAIN_CATEGORY:='%'||:BOOKS.MAIN_CATEGORY||'%';
        :BOOKS.SUB_CATEGORY:='%'||:BOOKS.SUB_CATEGORY||'%';
        :BOOKS.BOOK_TITLE:='%'||:BOOKS.BOOK_TITLE||'%';
        :BOOKS.EDITION:='%'||:BOOKS.EDITION||'%';

    elsif :BOOKS.c=3 then
        :BOOKS.BOOK_ID:=:BOOKS.BOOK_ID||'%';
        :BOOKS.MAIN_CATEGORY:=:BOOKS.MAIN_CATEGORY||'%';
        :BOOKS.SUB_CATEGORY:=:BOOKS.SUB_CATEGORY||'%';
        :BOOKS.BOOK_TITLE:=:BOOKS.BOOK_TITLE||'%';
        :BOOKS.EDITION:=:BOOKS.EDITION||'%';
```

```
    elsif :BOOKS.c=4 then
        :BOOKS.BOOK_ID:='%'||:BOOKS.BOOK_ID;
        :BOOKS.MAIN_CATEGORY:='%'||:BOOKS.MAIN_CATEGORY;
        :BOOKS.SUB_CATEGORY:='%'||:BOOKS.SUB_CATEGORY;
        :BOOKS.BOOK_TITLE:='%'||:BOOKS.BOOK_TITLE;
        :BOOKS.EDITION:='%'||:BOOKS.EDITION;
    end if;

elsif :global.tab=7 then

    if :AFFILIATED_ORGANIZATIONS.c=2 then
        :AFFILIATED_ORGANIZATIONS.ORG_NO:='%'||:AFFILIATED_ORGANIZATI
        ONS.ORG_NO||'%';
        :AFFILIATED_ORGANIZATIONS.AFF_ORGANIZATION:='%'||:AFFILIATED_
        ORGANIZATIONS.AFF_ORGANIZATION||'%';
        :AFFILIATED_ORGANIZATIONS.PUBLISHER:='%'||:AFFILIATED_ORGANIZ
        ATIONS.PUBLISHER||'%';
        :AFFILIATED_ORGANIZATIONS.ABBREVIATION:='%'||:AFFILIATED_ORG
        ANIZATIONS.ABBREVIATION||'%';

    elsif :AFFILIATED_ORGANIZATIONS.c=3 then
        :AFFILIATED_ORGANIZATIONS.ORG_NO:=:AFFILIATED_ORGANIZATION
        S.ORG_NO||'%';
        :AFFILIATED_ORGANIZATIONS.AFF_ORGANIZATION:=:AFFILIATED_OR
        GANIZATIONS.AFF_ORGANIZATION||'%';
        :AFFILIATED_ORGANIZATIONS.PUBLISHER:=:AFFILIATED_ORGANIZATI
        ONS.PUBLISHER||'%';
        :AFFILIATED_ORGANIZATIONS.ABBREVIATION:=:AFFILIATED_ORGANI
        ZATIONS.ABBREVIATION||'%';

    elsif :AFFILIATED_ORGANIZATIONS.c=4 then
```

```
        :AFFILIATED_ORGANIZATIONS.ORG_NO:='%'||:AFFILIATED_ORGANIZATI
        ONS.ORG_NO;
        :AFFILIATED_ORGANIZATIONS.AFF_ORGANIZATION:='%'||:AFFILIATED_
        ORGANIZATIONS.AFF_ORGANIZATION;
        :AFFILIATED_ORGANIZATIONS.PUBLISHER:='%'||:AFFILIATED_ORGANIZ
        ATIONS.PUBLISHER;
        :AFFILIATED_ORGANIZATIONS.ABBREVIATION:='%'||:AFFILIATED_ORG
        ANIZATIONS.ABBREVIATION;
    end if;

elsif :global.tab=8 then

    if :NEWSLETTERS.c=2 then
        :NEWSLETTERS.NEWSLETTER_NO:='%'||:NEWSLETTERS.NEWSLETTER_N
        O||'%';
        :NEWSLETTERS.NEWSLETTER:='%'||:NEWSLETTERS.NEWSLETTER||'%';

    elsif :NEWSLETTERS.c=3 then
        :NEWSLETTERS.NEWSLETTER_NO:=:NEWSLETTERS.NEWSLETTER_NO||'%';
        :NEWSLETTERS.NEWSLETTER:=:NEWSLETTERS.NEWSLETTER||'%';

    elsif :NEWSLETTERS.c=4 then
        :NEWSLETTERS.NEWSLETTER_NO:='%'||:NEWSLETTERS.NEWSLETTER_NO;
        :NEWSLETTERS.NEWSLETTER:='%'||:NEWSLETTERS.NEWSLETTER;
    end if;

elsif :global.tab=9 then

    if :PROCEEDINGS.c=2 then
        :PROCEEDINGS.PROCEEDING_NO:='%'||:PROCEEDINGS.PROCEEDING_NO|
        |'%';
        :PROCEEDINGS.PROCEEDING:='%'||:PROCEEDINGS.PROCEEDING||'%';
```

```
    elsif :NEWSLETTERS.c=3 then
        :PROCEEDINGS.PROCEEDING_NO:=:PROCEEDINGS.PROCEEDING_NO||'%';
        :PROCEEDINGS.PROCEEDING:=:PROCEEDINGS.PROCEEDING||'%';

    elsif :NEWSLETTERS.c=4 then
        :PROCEEDINGS.PROCEEDING_NO:='%'||:PROCEEDINGS.PROCEEDING_NO;
        :PROCEEDINGS.PROCEEDING:='%'||:PROCEEDINGS.PROCEEDING;
    end if;

elsif :global.tab=10 then

    if :SPECIAL_INTEREST_GROUPS.c=2 then
        :SPECIAL_INTEREST_GROUPS.SIG_NO:='%'||:SPECIAL_INTEREST_GROUP
        S.SIG_NO||'%';
        :SPECIAL_INTEREST_GROUPS.SIG:='%'||:SPECIAL_INTEREST_GROUPS.SIG
        ||'%';

    elsif :SPECIAL_INTEREST_GROUPS.c=3 then
        :SPECIAL_INTEREST_GROUPS.SIG_NO:=:SPECIAL_INTEREST_GROUPS.SI
        G_NO||'%';
        :SPECIAL_INTEREST_GROUPS.SIG:=:SPECIAL_INTEREST_GROUPS.SIG||'%';

    elsif :SPECIAL_INTEREST_GROUPS.c=4 then
        :SPECIAL_INTEREST_GROUPS.SIG_NO:='%'||:SPECIAL_INTEREST_GROUP
        S.SIG_NO;
        :SPECIAL_INTEREST_GROUPS.SIG:='%'||:SPECIAL_INTEREST_GROUPS.SIG;
    end if;

elsif :global.tab=11 then
```

```
if :TRANSACTIONS.c=2 then
    :TRANSACTIONS.TRANSACTION_NO:='%'||:TRANSACTIONS.TRANSACTIO
    N_NO||'%';
    :TRANSACTIONS.TRANSACTION_TITLE:='%'||:TRANSACTIONS.TRANSAC
    TION_TITLE||'%';
    :TRANSACTIONS.ABBREVIATION:='%'||:TRANSACTIONS.ABBREVIATION||
    '%';

elsif :TRANSACTIONS.c=3 then
    :TRANSACTIONS.TRANSACTION_NO:=:TRANSACTIONS.TRANSACTION_
    NO||'%';
    :TRANSACTIONS.TRANSACTION_TITLE:=:TRANSACTIONS.TRANSACTIO
    N_TITLE||'%';
    :TRANSACTIONS.ABBREVIATION:=:TRANSACTIONS.ABBREVIATION||'%';

elsif :TRANSACTIONS.c=4 then
    :TRANSACTIONS.TRANSACTION_NO:='%'||:TRANSACTIONS.TRANSACTIO
    N_NO;
    :TRANSACTIONS.TRANSACTION_TITLE:='%'||:TRANSACTIONS.TRANSAC
    TION_TITLE;
    :TRANSACTIONS.ABBREVIATION:='%'||:TRANSACTIONS.ABBREVIATION;
end if;

elsif :global.tab=12 then

if :MAGAZINES.c=2 then
    :MAGAZINES.MAGAZINE_NO:='%'||:MAGAZINES.MAGAZINE_NO||'%';
    :MAGAZINES.MAGAZINE:='%'||:MAGAZINES.MAGAZINE||'%';
    :MAGAZINES.ABBREVIATION:='%'||:MAGAZINES.ABBREVIATION||'%';
    :MAGAZINES.SOURCE:='%'||:MAGAZINES.SOURCE||'%';

elsif :MAGAZINES.c=3 then
```

```
        :MAGAZINES.MAGAZINE_NO:=:MAGAZINES.MAGAZINE_NO||'%';
        :MAGAZINES.MAGAZINE:=:MAGAZINES.MAGAZINE||'%';
        :MAGAZINES.ABBREVIATION:=:MAGAZINES.ABBREVIATION||'%';
        :MAGAZINES.SOURCE:=:MAGAZINES.SOURCE||'%';

    elsif :MAGAZINES.c=4 then
        :MAGAZINES.MAGAZINE_NO:='%'||:MAGAZINES.MAGAZINE_NO;
        :MAGAZINES.MAGAZINE:='%'||:MAGAZINES.MAGAZINE;
        :MAGAZINES.ABBREVIATION:='%'||:MAGAZINES.ABBREVIATION;
        :MAGAZINES.SOURCE:='%'||:MAGAZINES.SOURCE;
    end if;

end if;
execute_query;
```

- **WHEN_BUTTON_PRESSED on Push Button “RESET”:**

```
set_item_property('BOOKS.s',visible,property_true);
set_item_property('BOOKS.b',visible,property_true);
set_item_property('ARTICLES.TIT',visible,property_true);
:search_blk.sql:= ' ';
IF :SYSTEM.MODE='ENTER-QUERY' THEN
    EXIT_FORM;
ELSE
    clear_block;
END IF;
enter_query;
```

- **WHEN_BUTTON_PRESSED on Push Button “SHOW_SQL”:**

```
:search_blk.sql:=:system.last_query;
```

- **WHEN_BUTTON_PRESSED on Push Button “EXIT”:**

```
if :system.mode='ENTER-QUERY' then
    exit_form;
end if;
exit_form;
```

- List of Values (LOVs) and Record Groups:

- **TRAN_LOV:**

```
SELECT ALL TRANSACTIONS.TRANSACTION_TITLE,
TRANSACTIONS.TRANSACTION_NO, TRANSACTIONS.ABBREVIATION
FROM TRANSACTIONS
order by TRANSACTIONS.TRANSACTION_TITLE;
```

- **CONF_LOV:**

```
SELECT ALL CONFERENCES.CONFERENCE_NAME
FROM CONFERENCES
order by CONFERENCES.CONFERENCE_NAME;
```

- **CONF_MTG_YEAR_LOV:**

```
SELECT DISTINCT CONFERENCES.MTG_YEAR
FROM CONFERENCES
ORDER BY CONFERENCES.MTG_YEAR;
```

- **JOUR_LOV:**

```
SELECT DISTINCT JOURNALS.JOURNAL
FROM JOURNALS
ORDER BY  JOURNALS.JOURNAL;
```

- **BOOKS_LOV:**

```
SELECT DISTINCT BOOKS.BOOK_TITLE
FROM BOOKS
WHERE SUB_CATEGORY LIKE NVL(:BOOKS.SUB_CATEGORY,'%')
ORDER BY BOOKS.BOOK_TITLE;
```

- **BOOKS_MAIN_LOV:**

```
SELECT DISTINCT BOOKS.MAIN_CATEGORY
FROM BOOKS
ORDER BY BOOKS.MAIN_CATEGORY;
```

- **BOOKS_SUB_LOV:**

```
SELECT DISTINCT BOOKS.SUB_CATEGORY
FROM BOOKS
WHERE MAIN_CATEGORY LIKE NVL(:MAIN_CATEGORY,'%')
```

```
ORDER BY BOOKS.SUB_CATEGORY;
```

- **ART_AUTH_LOV:**

```
SELECT distinct ARTICLES.AUTHORS
FROM ARTICLES
WHERE ARTICLES.ARTICLE_TITLE LIKE
  NVL(:ARTICLES.ARTICLE_TITLE,'%')
order by ARTICLES.AUTHORS;
```

- **ART_TIT_LOV:**

```
SELECT distinct ARTICLES.ARTICLE_TITLE
FROM ARTICLES
order by ARTICLES.ARTICLE_TITLE;
```

- **ART_YR_LOV:**

```
SELECT DISTINCT ARTICLES.ARTICLE_YEAR
FROM ARTICLES
ORDER BY ARTICLES.ARTICLE_YEAR;
```

- **EDU_COURS_LOV:**

```
SELECT DISTINCT EDUCATIONAL_COURSES.COURSE_TITLE
FROM EDUCATIONAL_COURSES
ORDER BY EDUCATIONAL_COURSES.COURSE_TITLE;
```

- **MAG_LOV:**

```
SELECT DISTINCT MAGAZINES.MAGAZINE
FROM MAGAZINES
ORDER BY MAGAZINES.MAGAZINE;
```

- **NEW_LOV:**

```
SELECT DISTINCT NEWSLETTERS.NEWSLETTER
FROM NEWSLETTERS
ORDER BY NEWSLETTERS.NEWSLETTER;
```

- **PROC_LOV:**

```
SELECT DISTINCT  PROCEEDINGS.PROCEEDING
FROM PROCEEDINGS
ORDER BY PROCEEDINGS.PROCEEDING;
```

- **CIT_LOV:**

```
SELECT DISTINCT CITATIONS.CITATION
FROM CITATIONS
WHERE CITATIONS.AUTHORS LIKE NVL(:CITATIONS.AUTHORS, '%')
ORDER BY CITATIONS.CITATION;
```

- **CIT_AUTH_LOV:**
  SELECT DISTINCT CITATIONS.AUTHORS
  FROM CITATIONS
  WHERE CITATIONS.CITATION LIKE NVL(:CITATIONS.CITATION,'%')
  ORDER BY CITATIONS.AUTHORS;

- **CIT_YR_LOV:**
  SELECT DISTINCT CITATIONS.CITATION_YEAR
  FROM CITATIONS
  ORDER BY CITATIONS.CITATION_YEAR;

- **AFF_ORG_LOV:**
  SELECT DISTINCT AFFILIATED_ORGANIZATIONS.AFF_ORGANIZATION
  FROM AFFILIATED_ORGANIZATIONS
  ORDER BY AFFILIATED_ORGANIZATIONS.AFF_ORGANIZATION;

- **AFF_PUB_LOV:**
  SELECT DISTINCT AFFILIATED_ORGANIZATIONS.PUBLISHER
  FROM AFFILIATED_ORGANIZATIONS
  WHERE AFFILIATED_ORGANIZATIONS.AFF_ORGANIZATION LIKE
    NVL(:AFFILIATED_ORGANIZATIONS.AFF_ORGANIZATION,'%')
  ORDER BY AFFILIATED_ORGANIZATIONS.PUBLISHER;

- **SIG_LOV:**
  SELECT DISTINCT SPECIAL_INTEREST_GROUPS.SIG
  FROM SPECIAL_INTEREST_GROUPS
  ORDER BY SPECIAL_INTEREST_GROUPS.SIG;