Diplomarbeit

Putting the World Wide Web into a Data Warehouse:

A DWH-based Approach to Web Analysis

eingereicht von

Oliver David Witvoet

zur Erlangung des akademischen Grades
Magister rerum socialium oeconomicarumque
(Mag.rer.soc.oec.)
Magister der Sozial- und Wirtschaftswissenschaften

Fakultät für Wirtschaftswissenschaften und Informatik,
Universität Wien
Fakultät für Technische Naturwissenschaften und Informatik,
Technische Universität Wien

Studienrichtung: Wirtschaftsinformatik

Begutachter:
ao.Univ.Prof. Mag. Dr. Dieter Merkl
Univ. Ass. Dipl.-Ing. Dr. Andreas Rauber

Wien, im November 2002

Meinen Eltern

Abstract

The World Wide Web, due to its sheer size and dynamics, has turned into one of the most fascinating and important data sources for large-scale analysis and investigation, ranging from content-based information location, dynamics of change, to community analysis. Still, most projects rely on special-purpose tools optimized for a given task.

In this thesis I propose a Data Warehouse based approach to analyze the World Wide Web. Information contained in the Web pages, meta-data on the documents, as well as information acquired from additional sources such as the WHOIS database, are integrated into a multidimensional view of the Web. The resulting system allows for flexible analysis of the various characteristics of the Web. Results from a prototypical study of the Austrian national Web space as part of the AOLA project demonstrate the potential of the presented approach.

Zusammenfassung

Das World Wide Web, mit seiner Größe und Dynamik, hat sich als eine der faszinierendsten und wichtigsten Datenquellen für die Erforschung und Analyse in großem Umfang, angefangen von der Auswertung des Inhalts ,über die Dynamik der Veränderung bis hin zur so genannten 'Community' Analyse, herausentwickelt. Bis jetzt beschränken sich die meisten Projekte auf spezielle Tools, die für eine bestimmte Aufgabe entwickelt worden sind.

In dieser Diplomarbeit stelle ich einen Ansatz vor, das World Wide Web mit Hilfe eines Data Warehouses zu analysieren. Informationen, die die einzelnen Webseiten beinhalten, Metadaten der Dokumente, sowie weitere Informationen, die von zusätzlichen Quellen, wie zum Beispiel der WHOIS Datenbank abgefragt werden, sind in einer multidimensionalen Sichtweise des Webs integriert. Das resultierende System erlaubt eine flexible Analyse der verschiedenen Charakteristiken des Webs. Das Potenzial des präsentierten Ansatzes wird durch die Ergebnisse der Untersuchung des Österreischen Webs als Teil des AOLA Projekts eindrucksvoll demonstriert.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The World Wide Web is a popular and interactive medium which changes every day, even every second. It is the biggest, most versatile, and thus one of the most challenging data repositories to 'manage'. In the past years we have not only seen an incredible growth of information available on the Web, but also a shift of the Web from a platform for distributing information among IT-related persons to a general platform for all levels of society. It is being used as a source of information and entertainment, forms the basis for e-government, and e-commerce, has inspired new forms of art, and serves as a general platform for meeting and communicating via various discussion forums. By now it attracts and involves a broad range of groups in our society, from school children, professionals of various disciplines, up to seniors. They are all forming their communities on the Web, consuming and sharing experience, using it for transactions in different ways. At the same time technology keeps advancing, providing new forms of interaction, new designs of portals. Finding documents regarding a special topic is relatively easy. This process of searching for valuable information in the Web is called resource discovery (RD). In a typical RD procedure, the user submits a query Q, which is simply a list of keywords (probably with some additional parameters), to the RD server. As a result, the server returns a set of related Web pages. To make this procedure available for the Internet users, there are many search engines, such as Yahoo!, AltaVista, Google, Excite, Hotbot, Infoseek, alltheweb, etc. In general, a search engine collects Web pages on the Internet

through a robot (crawler) program. Then these Web pages are automatically scanned to build giant indices, in order to retrieve quickly the set of all Web pages containing the given keywords. Yet, the problem is that the number of retrieved sites is too large and they vary widely in quality, also referred to as the *Abundance Problem* [8]. In particular, a topic of any breadth will typically contain several thousands or millions of more or less relevant Web pages. For instance, if one queries the search engine AltaVista, for the phrase 'data mining', over 50 000 Web pages will be found. A user typically wants to look at just a few of them. The ranking algorithm discussed in Section 3.2 tries to give the most important Web pages on top of the result list.

Yet, there is significantly more to the Web than the mere content available at various Web sites. The graphs formed by the hyperlinked documents provide information about Web connectivity and communities active in certain topic areas. Information gathered from Web servers provides information of market penetration of various technologies and vendor products. By collecting information from the Web over a certain period of time, information about technology evolution and Web usage can be deducted, such as the advent of interactive Web sites, encryption for e-commerce transactions, required critical mass for successful technology deployment, and so on. Because of these challenges Web mining, which is the process of applying data mining techniques to the discovery of patterns from Web data, has received much interest recently.

In this thesis I am presenting the *Austrian On-Line Archive Processing* project*(AOLAP)*, a very large Data Warehouse in order to provide a comprehensive analysis of the Austrian Web space. Within this project we take an analytical view of the Web as a data and technology repository evolving over time. With such a repository of Web data, as well as the meta-data, which is associated with the documents and domains, we have a powerful source of information that goes beyond the content of Web pages. The Web is not only content, but technically speaking, rather a medium for transporting content in various ways, using various technical platforms as well as data representations to make its information available. The providers of information are located in different physical places on the hyperlinked world, and information is transferred via a variety of channels. Having an archive of the World Wide Web means

that we can not only see which information was available at a certain time, but also trace where information was being produced and replicated, which technology has been used for representing a certain kind of information, what kind of systems has been used to make the information available.

The answers to these kind of questions require a different perspective of the Web and Web archives, focusing not solely on content, but on the wealth of information automatically associated with each object on the Web, such as its file format; its size and the recentness of its last update; its link structure and connectivity to other pages within the same site, domain, and externally; the language used; the operating system and Web server software running on the server side machine; the physical location of the machine; the geographical distribution and saturation of Web information services; the use of specific protocols; cookies; and many more.

It also gives us the means to trace the life cycle of technology, following file formats, interaction standards, and server technology from their creation, via different degrees of acceptance to either prolonged utilization or early obsolescence. It provides a basis for tracking the technological evolution of different geographical areas, analyzing characteristics such as the "digital divide", not only from a consumer's point of view, i.e. who has access to Web information, and who has not, but also from a provider's point of view, i.e. which areas in the world, as well as on a much smaller, regional scale, are able to make themselves heard, are able to participate in the exchange of information by publishing information on their own account on the Web.

The process of building a Data Warehouse can be broken down into three phases: gathering data, data transformation, and filling the database. We built three independent modules to be able to separate these three phases. The first module specifies the acquisition of the data. As the primary source of information we use the data gathered by the *Austrian On-Line Archive (AOLA)* project, as well as additional information retrieved from sources such as the WHOIS server. In the context of World Wide Web, hypertext and linking together of related pieces of information has become a real problem. Hypertext documents are written by multiple, independent authors who can create links between pages with indiscretion. Navigational links, citation links, reference

links, press links, financial links and just plain confusing links are creatively mixed together and scattered throughout the pages. Consistency is rarely found within a Web site. In order to incorporate link structure analysis into a Data Warehouse, considerable design efforts were required to map the generic graph structure in a Data warehouse (DWH) model.

The remainder of this thesis is structured as follows. In the following Chapter 2 I will describe the taxonomy of the *Web mining* in general. Web mining can be divided into three sub-groups namely *Web Contenct Mining, Web Usage Mining, and Web Structure Mining*. In that chapter a brief overview of these definitions will be given. Further in Chapter 3 we will take a closer look to the works related to our project. Chapter 4 deals with the data acquisition, transformation, and adjusting to a useful and simple structure for our project. This process is divided into three different modules, which I will describe in that chapter. In Chapter 5 I present the schema of the database describing the several tables. For a better understanding, screenshots of the most important tables and connections of the DB-schema are shown. The feeding process of the database is presented in seven different steps in Chapter 6. The Data Warehouse including its cubes and its dimensions are presented in Chapter 7. In Chapter 8 I will introduce the OLAP tool we used in this project, as well as the program used to create the charts. The analysis of our experimental results are presented in Chapter 9. Additionally, a sample analysis process, beginning from the setting of the OLAP cube to the creation of different diagrams, is described step by step at the beginning of that chapter. In Chapter 10 I will briefly describe the hardware and software used in the project. Final conclusions are provided in Chapter 11. The appendix A provides a full list of the Web server types gathered.

Chapter 2

Web Mining taxonomy

In this chapter I will describe the data mining efforts associated with the Web, called *Web mining*. It can be broadly divided into three classes, namely *content mining, usage mining,* and *structure mining*, described in the following sections.

2.1 Web Content Mining

The heterogeneity and the lack of structure that permeates much of the ever expanding information sources on the World Wide Web, such as hypertext documents, makes automated discovery, organization, and management of Web-based information difficult. Traditional search and indexing tools of the Internet and the World Wide Web such as Lycos, Alta Vista, WebCrawler, Google, MetaCrawler, and others, provide some comfort to users, but they do not generally provide structural information nor categorize, filter, or interpret documents.

In recent years these factors have prompted researchers to develop more intelligent tools for information retrieval, such as intelligent Web agents, as well as to extend database and data mining techniques to provide a higher level of organization for semi-structured data available on the Web. Some of these efforts are described below.

Agent-Based Approach The agent-based approach to Web mining involves the development of sophisticated AI systems that can act autonomously or semi-

autonomously on behalf of a particular user, to discover and organize Web-based information. Generally, the agent-based Web mining systems can be placed into the following three categories:

Intelligent Search Agents Several intelligent Web agents have been developed that search for relevant information using characteristics of a particular domain (and possibly a user profile) to organize and interpret the discovered information. For example, agents such as Harvest [15], FAQ-Finder [16], Information Manifold [17], OCCAM [18], and ParaSite [19] rely either on prespecified and domain specific information about particular types of documents, or on hard coded models of the information sources to retrieve and interpret documents. Other agents, such as ShopBot [20] and ILA (Internet Learning Agent) [21], attempt to interact with and learn the structure of unfamiliar information sources. ShopBot retrieves product information from various vendor sites using only general information about the product domain. ILA, on the other hand, learns models of various information sources and translates these into its own internal concept hierarchy.

Information Filtering/Categorization There are a number of Web agents who also use various information retrieval techniques [22] and characteristics of open hypertext Web documents to automatically retrieve, filter, and categorize them. For example, HyPursuit [23] uses semantic information embedded in link structures as well as document content to create cluster hierarchies of hypertext documents, and to structure an information space. Prospectminer [43] attempts to identify *sales prospects* based on the initial query information provided by the user. It generates the information by searching through the hypertexts of company Web sites, press releases, etc. At the next search the Prospectminer will take the feedback of the user into account.

Personalized Web Agents Another category of Web agents are those that obtain or learn user preferences and discover Web information sources that correspond to these preferences, and possibly those of other individuals with similar interests (using collaborative filtering). A few recent examples of such

agents include the WebWatcher [24], PAINT [25], and Syskill & Webert [26]. For example, Syskill & Webert is a system that utilizes a user profile and learns to rate Web pages of interest using a Bayesian classifier[1].

Database Approach The database approaches to Web mining have generally focused on techniques for integrating and organizing the heterogeneous and semi-structured data on the Web into more structured and high-level collections of resources, such as in relational databases, and using standard database querying mechanisms and data mining techniques to access and analyze this information.

Multilevel Databases Several researchers have proposed a multilevel database approach organizing Web-based information. The main idea behind these proposals is that the lowest level of the database contains primitive semi-structured information, stored in various Web repositories, such as hypertext documents. At the higher level(s) meta-data or generalizations are extracted from lower levels and organized in structured collections such as relational or object-oriented databases. For example, Han, et. al. [27] use a multi-layered database where each layer is obtained via generalization and transformation operations performed on the lower layers. Kholsa, et. al. [28] propose the creation and maintenance of meta-databases using a global schema. King & Novak [29] propose the incremental integration of a portion of the schema from each information source, rather than relying on a global heterogeneous database schema. The ARANEUS system [30] extracts relevant information from hypertext documents and integrates these into higher-level derived Web Hypertexts, which are generalizations of the notion of database views.

Web Query Systems There have been many Web-based query systems and languages developed recently that attempt to utilize standard database query languages such as SQL, structural information about Web documents, and even natural language processing for accommodating the types of queries that are used in World Wide Web searches. For example, the Web-based query

[1] A Bayesian Classifier is a computer program, which uses statistical properties of the available data to develop a classification model based on the Bayes Law of probability.

systems W3QL [31] combines structure queries, based on the organization of hypertext documents, and content queries, based on information retrieval techniques.

2.2 Web Usage Mining

Web usage mining is the application of data mining techniques to discover patterns from Web data, in order to understand and better serve the needs of Web-based applications. As more organizations rely on the Internet and the World Wide Web to conduct business, the traditional strategies and techniques for market analysis need to be revised in this context.

Web usage mining consists of three phases, namely *preprocessing, pattern discovery,* and *pattern analysis.*

Preprocessing is the process of transforming and adjusting the data acquired to a useful structure.

Pattern Discovery Tools The emerging tools for user pattern discovery use sophisticated techniques from AI, data mining, psychology, and information theory, to mine for knowledge from collected data. For example, the WEBMINER system [12] introduces a general architecture for Web usage mining. WEBMINER automatically discovers association rules and sequential patterns from server access logs. For more information about these server access logs, please refer to the paragraph *Web Log File Mining* beyond. These can in turn be used to perform various types of user traversal path analysis such as identifying the most traversed paths thorough a Web locality. Further it is possible to combine path traversal patterns, Web page typing, and site topology information to categorize pages for easier access by users.

Pattern Analysis Tools Once access patterns have been discovered, analysts need the appropriate tools and techniques to understand, visualize, and interpret these patterns. Examples of such tools include the WebViz system [13] for visualizing path traversal patterns. Others have proposed using OLAP techniques such as data cubes for the purpose of simplifying the analysis of usage

statistics from server access logs [14]. The WEBMINER system proposes an SQL-like query mechanism for querying the discovered knowledge (in the form of association rules and sequential patterns).

Another pattern analysis tool is *WUM: Web Utilization Miner* [40]. WUM is a sequence miner. Its primary purpose is to analyze the navigational behaviour of users in a Web site, but it is appropriate for sequential pattern discovery in any type of log. It discovers patterns comprised of not necessarily adjacent events and satisfying user-specific criteria. WUM is an integrated environment for log preparation, querying and visualization. Its mining query language MINT supports the specification of criteria describing dominant or statistically rare patterns. Its visualization mechanism displays the nodes comprising the desired pattern and the different non-frequent paths located in-between. This mechanism is important in order to examine how the Web site is really being navigated.

Web Log File Mining Organizations often generate and collect large volumes of data in their daily operations. Most of this information is usually generated automatically by Web servers and collected in server access logs. Other sources of user information include referrer logs, which contains information about the referring pages for each page reference, and user registration or survey data gathered via tools such as CGI scripts.

Analyzing such data can help these organizations to determine the life time value of customers, cross marketing strategies across products, and effectiveness of promotional campaigns, among other things. Analysis of server access logs and user registration data can also provide valuable information to create better structured Web sites and hence, get a more effective presence. In organizations using intranet technologies, such analysis can shed light on more effective management of workgroup communication and organizational infrastructure. Finally, for organizations that sell advertising on the World Wide Web, analyzing user access patterns help assigning ads to specific groups of users.

Most of the existing Web analysis tools provide mechanisms for reporting user activity in the servers and various forms of data filtering. Using such tools, for example, it is possible to determine the number of accesses to the server and

the individual files within the organization's Web space, the frequency and time intervals of visits, and domain names, and the URLs of users of the Web server. In general, these tools are designed to deal with low to moderate traffic servers, and furthermore, they usually provide little or no analysis of data relationships among the accessed files and directories within the Web space.

More sophisticated systems and techniques for discovery and analysis of patterns are emerging.

2.3 Web Structure Mining

Web structure mining is the process of inferring knowledge from the World Wide Web organization and links between references and referents in the Web. The World Wide Web reveals more information than just the information contained in documents. For example, links pointing to a document indicate the popularity of the document, while links coming out of a document indicate the richness or perhaps the variety of topics covered in the document. There are some projects concentrating on mining the linkage structure of the Web resources, like the WHOWEDA project described in Section 3.1.5. Different ranking algorithms are implemented to perform a clever ranking of Web sites. Some of the most famous approaches are described in Section 3.2. Due to this knowledge it is possible to create a ranking of Web sites in order to find the 'valuable' sites. This improved type of content retrieval of the Web is already used by several search engines. The first famous one using this technic was Google (`http://www.google.com/`).

2.4 Summing-up: Web Mining

Web mining is the process of finding useful information and patterns (data mining) concerning the WWW. The results of these analysis can help to access and design the WWW more efficiently. An efficient WWW combined with an easy access to a broader range of users is of advantage for the whole economy and, as a consequence, for all consumers.

In this chapter I described the three subgroups of *Web Mining*. *Web Content*

mining, which describes the process of getting the essence from within Web pages is described in Section 2.1. *Web usage mining*, which is the analysis of the Web usage data like e.g. the Web log files of a Web server, is described in Section 2.2. *Web Structure mining* described in Section 2.3, deals with the analysis of the linkage structure of the Web.

Chapter 3

Related work

Numerous projects deal with the collection and the discovery of patterns from Web data, including the wealth of search engines performing information retrieval, clustering search results into consistent subgroups to assist in navigation, automatic text categorization tools filing documents into pre-defined categories, and a wealth of other types of content-related analysis. Furthermore, an equal share of projects addresses issues pertaining to Web analysis, to improve retrieval performance, or to detect communities on the Web.

In Section 3.1 I will provide a brief description of the most famous Web archiving projects. In Section 3.2 we will talk about ranking algorithm of the Web. Furthermore I will describe some projects using such algorithms. Some projects evaluating statistics about the Internet are briefly described in Section 3.3. In that section I will also provide a few numbers about the Internet to visualize the bigness and discrepancy of the Web.

3.1 Web Archiving Projects

With the popularity of the Internet it has become a part of our cultural heritage. For this reason as well as for economical reasons there is an upcoming interest to analyze the evolution of Web. This can only be achieved when the data in the Internet is stored permanently. Therefore we find numerous projects aiming at creating large-scale repositories containing excerpts and snapshots of Web data.

Some of these are described in this section below.

One of the biggest digital libraries of Internet sites and other cultural artifacts in digital form is the *Internet Archive* [41], which provides free access to researchers, historians, scholars, and the general public. Founded in 1996 and located in the Presidio of San Francisco, the Archive has reached a size of more then 100 Terrabyte of data, donated by the search engine Alexa, which are accessible online.

Another Web page repository is being built within the *WebBase* project at Stanford University, addressing issues such as the functional design, storage management, as well as indexing modules for Web repositories [36]. The main goal of this project is to acquire and store locally a subset of a given Web space in order to facilitate the performant execution of several types of analyses and queries, such as page ranking, and information retrieval. However, it limits its scope with archiving of one copy of each page at a time, thus providing no historization. It focuses on HTML pages only.

With respect to the usage of these Web archives, the *Nordic Web Archive* initiative [3] is currently developing an access interface, that will allow users to search and surf within such an archive. A similar interface, called the *Wayback Machine*, is already available for the *Internet Archive* described above, providing, for each URL entered, a timeline listing the dates when this specific URL was added to the archive, i.e. which versions of the respective files are available.

3.1.1 The Austrian On-Line Archive (AOLA)

The *Austrian On-Line Archive*[1] (AOLA) [11] is an initiative to create a permanent archive documenting the rise of the Austrian Internet, capturing the sociological and cultural aspects of the Austrian Web space. With respect to the AOLA project, the Austrian Web space covers the whole '.at' domain, but also servers located in Austria yet registered under "foreign" domains like *.com, .org, .cc,* etc. are included. The inclusion of these servers so far is determined semi-automatically by maintaining a list of allowed non-at servers. Furthermore, sites dedicated to topics of Austrian interest as well as sites about Austria

[1] `http://www.ifs.tuwien.ac.at/~aola`

(so-called "Austriaca") are considered even if they are physically located in another country. Austrian representations in a foreign country like the Austrian Cultural Institute in New York City at `http://www.aci.org/`, are examples for such sites of interest. These sites are fed into the system using a currently manually maintained list. Web crawlers, specifically the Combine Crawler, are used to gather the data from the Web. While the crawling process itself runs completely automatically, manual supervision and intervention is required in some cases when faulty URLs are encountered. The pages downloaded from the Web are stored together with additional meta-data in a hierarchical structure defined by the Sweden's *Kulturaw3*-project, and archived in compressed format on tapes.

The data and the associated meta-data gathered from the crawl by the AOLA project are the basis for our analysis within the AOLAP project. The archive currently consists of about 488 GB of data from two crawls, with more than 2,8 million pages from about 45.000 sites from the first partial crawl in 2001 (118 GB in total), as well as about 370 GB (approx. 13 Mio URLs from more than 133.400 different Web sites, which amounts to about 184.000 sites including alias names of servers) from the second crawl in spring 2002.

3.1.2 Kulturaw3 Project

Within Europe, the leading project with respect to Web archiving is the *Kulturaw3* project by the Swedish Royal National Library [2]. Its archive contains frequent snapshots of the Swedish national Web space starting in 1996, using the *Combine* harvester as their means of data acquisition. Initially, this tool was designed for indexing purposes by the University of Lund, Sweden, in the scope of the DESIRE-project founded by the European Commission. This crawler is very flexible, allowing intervention while the system is running.

3.1.3 NEDLIB Project

A second large initiative in this field is the archiving initiative of the *NEDLIB* project [7, 35], headed by the Finnish National Library and the Helsinki Center

for Scientific Computing. Within the scope of the project, a special crawler specifically geared towards tasks of Web page archiving has been developed, and has been used to acquire a snapshot of the Finnish Web space. This tool has also been used by other national groups, e.g. in Iceland, to build collections of their respective Web space. Similar initiatives are being followed in the Czech Republic by the National Library at Brno, the National Libraries of Norway and Estonia, and others.

The first harvesting round of the Finnish Web space was completed in June 2002. The archive consists of 42 million different URLs. The most popular file formats of this archive are compared against the Swedish ones in Section 4.5. As described in Section 3.1.1 the base data we rely on is not only the data crawled from the '.at' domain but also Austrian servers with another top-level domain like '.com', '.org', or servers in foreign countries dealing with Austria. The Finnish project did quite the same, they have harvested not only the root domain '.fi', but also Finnish servers from other domains. They applied two complementary methods to do this. First, they co-operate with Info Center Finland; a company, which has maintained a portal of Finnish sites since 1997, where they got a large set of Finnish servers. Second, they used linguistic methods to track down Internet names with Finnish words in them. The resulting list was manually checked, weeded and then provided as an input to the harvester as well.

3.1.4 Web Archeology

Web Archeology [34] is a project that develops tools for the process of exploring the Web, locating promising sources of information, and sifting through data to discover relevant artifacts. These tools can be classified into different levels forming a pyramid, where the lowest layer of the pyramid comprises the tools for gathering data from the Web. Their primary data gathering tool is the *Mercator extensible Web crawler*, a high-performance Web crawler. Mercator is designed to scale to the entire Web. It has been used to fetch tens of millions of documents. Scalability is achieved by implementing the data structures so that they use a bounded amount of memory, regardless of the size of the crawl.

Marc Najork and Janet L. Wiener [10] give a more exact insight to the Crawling algorithm used in this project. The next layer is the storage system for the data gathered from the Web. The Web-in-a-box project is currently building a storage system capable of holding many billions of Web pages that provides an efficient, easy-to-use query-based interface for upper-level tools that access the data. Because of the large amount of data in Web Archaeology, special-purpose databases are used to optimize access to various features of Web data. The two main feature databases they have built are the *Connectivity Server* and the *Term-vector Database*. The Connectivity Server provides fast access to URLs and the links between them, while the Term-vector Database specializes in providing word frequency counts for text (including HTML) pages. The highest level of this pyramid are applications and end-user tools that help other people to use the Web as, for example, in the *Geodesy* project [5], trying to discover and measure the structure of the Web. The Web is presented as a graph where each URL (Web page) is a node and each link from URL A to URL B is a directed edge from node A to node B. By measuring the expected shape of the Web, it is then possible to spot anomalies in the Web graph. The anomalies, in turn, can help to spot spam, find families of related hosts, or develop new models for the Web.

3.1.5 The WHOWEDA Approach

WHOWEDA (Warehouse of Web Data) [9] is a meta-data repository of Web information, available for queries and analysis. Conventional search engine queries may be described as single-node queries; one specifies a set of keywords that are expected to be found in certain Web documents. The WHOWEDA database provides a structured hierarchy of topological querying; different sets of keywords may be specified on multiple nodes and additional criteria may be defined for the hyperlinks among the nodes. Thus, the query is a graph-like structure and it is used to match portions of the WWW satisfying its conditions. The information filling the database is coupled from various sources, translated into a common Web data model, and integrated with the existing data in WHOWEDA. WHOWEDA consists of two major components: a data manipulation module

called Web Information Coupling System (WICS) and a data mining module called Web Information Mining System (WIMS), which provides various forms of data mining and knowledge discovery. WICS extracts and retrieves the WWW information called Web tuples and Web schemas [4], and stores it into the warehouse. A Web tuple is a directed graph consisting of sets of node and link objects. A node represents the meta-data associated with a Web document and the content of the structure of the document as, for example, the URL, title, format, size, date, or date of last modification, and a node data tree to represent the content and structure of the document. A link consists of a set of link meta-data as, for example, the link type or the target URL of the link.

A Web schema in WHOWEDA provides two types of information: First, it specifies some of the common properties shared by the documents and hyperlinks in the query result (called Web table) as the structure, the meta-data as well as the content. The Web table is a collection of Web tuples. There is a schema associated with every Web table.

The Web schemas contain the meta-data and structural data of the Web content. Some of the nodes and links in a set of Web tuples may share common characteristics with respect to their content, structure, and meta-data. For instance, given a set of documents in a Web table, the Web schemas may specify that the title of all these documents contain the keyword *genetic disorder*. It may also specify that these documents belong to the Web site at `http://www.ninds.nih.gov` and contain the keywords 'symptom', 'treatment' and drugs inside the tag disease. Secondly a Web schema summarizes the hyperlink structure of a collection of interlinked Web documents (Web tables). For instance, if all nodes of type A (e.g. Web documents containing the same title) are directly connected to nodes of type B (e.g. Web documents of the same format), then the Web schema may express this connectivity property involving nodes of types A and B.

As our project, the WHOWEDA provides a WWW archive, which can be accessed when the Web is not reachable. Contrary to our project the WHOWEDA project contains Web structure data of a very high granulation (internal structure of the different documents). We decided to save just the structure among the Web documents. This is to put the focus on other, for us more important,

data. And for sure, as our project is in the starting period there are a lot of features of the WHOWEDA project, which we will implement in the future. Further we do not have a graphical front-end client for creation, retrieving, and manipulation of information as in WHOWEDA. But one of the biggest advantages of our project is the enormous flexibility to analyze every kind of pattern, anomalies, and distributions of the Web data.

3.2 Ranking Algorithm

For many topics, the World Wide Web contains hundreds or thousands of relevant documents of widely varying quality. It is hard to identify a small subset of specific documents. Ranking Algorithms [33] have received much interest recently, to identify high quality items out of the huge amount of data available.

At the present time, most rank algorithms of Web resources are using similarity measure based on vector-space model. To compute the similarities, we can view each document as a n-dimensional vector $< w_1, ..., w_n >$. The term w_i in this vector represents the i^{th} word in the vocabulary. If w_i does not appear in the document, then w_i is zero. If it does appear, w_i is set to represent the significance of the word. One common way to compute the significance w_i is to multiply the number of times the i^{th} word appears in the document by the inverse document frequency (IDF) of the i^{th} word. The IDF factor is divided by the number of times the word appears in the entire 'collection', which in this case would be the entire Web. The IDF factor corresponds to the content discriminating power of a word: a term that appears rarely in documents (e.g., 'algebra') has a high IDF, while a term that occurs in many documents (e.g., 'the') has a low IDF. The w_i terms can also take into account where in a HTML page the word appears, for instance, words appearing in the title may be given a higher weight than other words in the body. Along with the popularization of Web meta-data standards such as RDF (Resource Description Framework developed by the W3C), it becomes feasible to take advantage of the meta-data of Web resources, which can also improve the rank algorithm's accuracy.

But the standard rank algorithms have lots of limitation. They only eval-

uate the content, but totally neglect the quality of Web resources. So these rank algorithms can be easily cheated. Webmasters can make their sites highly ranked through inserting some irrelevant but popular words (e.g. 'Madonna', 'sex') into important places (e.g. page title) or meta-data. This phenomenon is called *Search Engine Persuasion (SEP)* or *Web Spamming*. This is in fact a big problem that search engines have to face. Indeed, search engines have become so important in the advertising market that it has become essential for companies to have their pages listed in top positions of search engines, in order to get a significant Web-based promotion. This phenomenon is boosting at such a rate as to have provoked serious problems for search engines, and has revolutioned the Web design companies, who are now specifically asked not only to design good Web sites, but also to make them rank high in search engines. A vast number of new companies was born just to make customer Web pages as visible as possible. More and more companies, like Exploit, Allwilk, Northern Webs, Ryley & Associates, etc., explicitly study ways to rank a page highly in search engines. OpenText arrived even to sell 'preferred listings' i.e., assuring that a particular entry will stay in the top ten for some time, a policy that has provoked some controversies.

This has led to a bad performance degradation of search engines, since an increasingly high number of pages is designed to have an artificially high textual content. The phenomenon is so serious that search engines like InfoSeek and Lycos have introduced penalties to face the most common of these persuasion techniques, 'spamdexing'; i.e, the artificial repetition of relevant keywords. Despite these efforts, the situation is getting worse and worse, since more sophisticated techniques have been developed, which analyze the behavior of search engines and tune the pages accordingly.

Recent researches in this area concentrate on mining the linkage structure of Web resources to support resource discovery in the Web. The typical one in such rank algorithms is PageRank described below in Section 3.2.4, which is proposed by the Stanford University and has been applied in the famous search engine Google (`http://www.google.com/`). Another approach for ranking Web sites is to calculate so called *Hubs* and *Authorities* described in the following paragraph.

Hubs and Authorities The links connecting documents in the Web are in principle all equivalent: the Web itself does not express a preference for one link or one document above another. Yet, the connectivity or pattern of linkages between pages does contain a lot of implicit information about the relative importance of links. The author of a Web document will normally only include links to other documents that are relevant to the general subject of the page, and of sufficient quality. Thus, locating one document relevant to your goals may be sufficient to guide you to further information on that issue. High quality documents that contain clear, accurate and useful information, are likely to have many links pointing to them, while low quality documents will get few or no links. Thus, although no explicit preference function is attached to a link, there is a preference implicit in the total number of links pointing to a document. This preference is produced collectively, by the group of all Web authors. Hubs and authorities exhibit a mutually reinforcing relationship: a good hub points to many good authorities; a good authority is pointed to by many good hubs. Whereas a single page can be both, a good authority as well as a good hub.

3.2.1 HITS (Hyperlinked Induced Topic Search) algorithm

HITS is an algorithm to calculate *hubs* and *authorities* in response to a query topic. HITS proceeds as follows:

1. Starting from a user-supplied query, HITS assembles an initial set of pages: typically, up to 200 pages returned by a text search engine such as AltaVista on that query. These pages are then expanded to a larger root set by adding any pages that are linked to or from any page in the initial set.

2. HITS then associates with each page p a hub-weight h(p) and an authority weight a(p), all initialized to 1.

3. HITS then iteratively updates the hub and authority weights of each page in the root set as follows. First, under the intuition that a page pointing to good authorities should be considered a good hub, we replace the hub

score of each page by the sum of the authorities of the pages it points to. And second, dually, under the intuition that a page pointed to by good hubs should be considered a good authority, we replace the authority score of each page by the sum of the hub scores of the pages pointing to it.

4. The update operations are performed for all the pages, and the process repeated (normalizing the weights after each iteration) for some number of rounds. Following this, the pages with the highest h(p) and a(p) scores are output as the best hubs and authorities.

3.2.2 Clever

As in the HITS project, the thesis underlying Clever, is to classify the content of the Web in two kinds of valuable pages, the *hubs* and *authorities*. Clever additionally uses the content of the Web pages, thus exploiting not only the link structure (as in HITS) but further using the text and other properties of the Web pages being distilled.

Given a traditional text query, such as the Altavista text search engine, Clever starts from an initial set of around 200 pages. It then expands the initial set to generate the root set by adding any page pointing to, or pointed to by, a page in the initial set; typically, the root set contains a few thousand pages. Afterwards, a graph is built where each node is a page of the root set. There are weighted directed edges from a node to another if the former has a hyperlink to the latter. The weight of the edge is a function of the relevance of the text surrounding the anchor, augmented by several information, for instance, whether the same author created both pages. Having created this graph, Clever then determines hubs and authorities by applying the basic HITS algorithm augmented with a number of additional data as mirrored pages, shared domain names, and so on.

3.2.3 Kleinberg's Algorithm for computing Hubs and Authorities

Also Kleinberg worked on the problem of receiving a small set of relevant authoritative results from a Web query. He formalized the quality of documents within a hyper-linked collection also using the concept of *hubs* and *authority*. Their approach is to define a root set S as follows. For a number k (say 200), they defined S to be the top k pages indexed by some term-based search engine. Then they grew it to a larger base set T, consisting of all pages that either belong to S, point to a page in S, or are pointed to by a page in S. Then they ranked the pages according to their in-degree or backlinks (number of links pointing to this site) in the subgraph induced by T. Authoritative pages relevant to the initial query should have large in-degree, and there should also be considerable overlap in the sets of pages pointing to them. These pages, which point to multiple relevant authoritative pages are the hubs. As I described above, hubs and authorities stand in a mutually reinforcing relationship: a good authority is a document that is linked to by many good hubs, and a good hub is a document that links to many authorities.

The final output of the algorithm calculating the authority weights and hub weights is a pair of sets (X,Y), where X is a small set of authorities and Y is a small set of hubs. This is the desired small set of high-quality pages that can be returned in response to a user query. Kleinberg calls this pair of sets (X,Y) a *community* of hubs and authorities, which are characterized by their mutually reinforcing relationship, which can be seen in Figure 3.1.

3.2.4 PageRank

PageRank is a global ranking of all Web pages, regardless of their content, based solely on their location in the Web's graph structure. As the *HITS* algorithm, the *PageRank* algorithm determines the quality or *authority* of a Web page on the basis of the number and quality of the pages that link to it. Since the definition is recursive (a page has high quality if many high quality pages point to it), the algorithm needs several iterations to determine the overall quality of

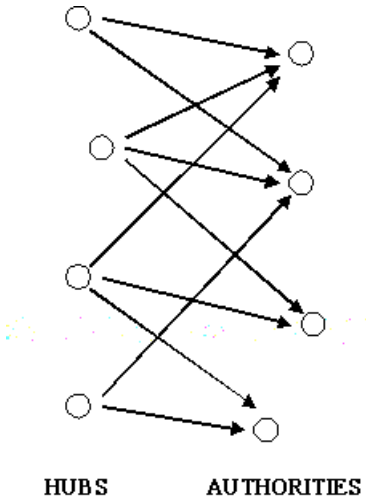


Figure 3.1: A dense community of hubs and authorities.

a page.

In combination with a keyword search, which restricts the pages for which the quality is computed to a specific problem, this method seems to produce a much better quality in the answers returned for a query. The disadvantage of this method, compared to the *learning Web algorithms* [42], is that it is static. This means they merely use the rather sparse linking pattern that already exists instead of allowing the Web to adapt to the way it is used, which the learning algorithm support.

PageRank [32] is another algorithm to measure the relative importance of Web pages. It is a method for computing a ranking for every Web page based on the graph of the Web described above. In simple words, this algorithm is based on the following assumption. A page has high rank if the sum of the ranks of its backlinks (number of links pointing to this site) is high. This covers both the case when a page has many backlinks and when a page has fewer highly ranked backlinks.

Technical definition of PageRank Every page has some number of forward links and backlinks, which is shown in Figure 3.2. Note that the rank of a page

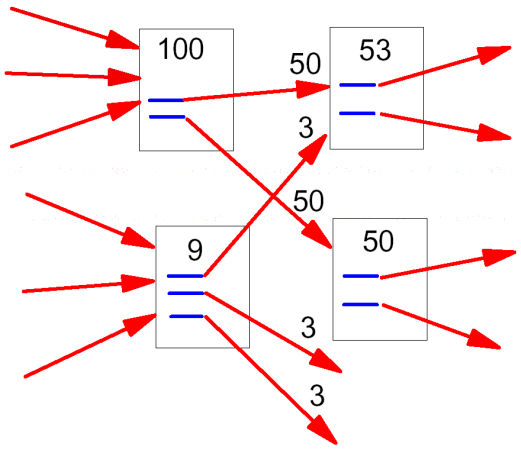


Figure 3.2: Simplified PageRank calculation

is divided among its forward links evenly to contribute to the ranks of the pages they point to. For example, if a page has a rank of 100 and contains 2 links, each site linked from this page 'gets' the rank of 50 from this page. Note that there are a number of pages with no forward links, therefore their weight is lost from the system. These links are called *dangling links*. Dangling links are simply links that point to any page with no outgoing links. Because dangling links do not affect the ranking of any other page directly, they are simply removed from the system. The process calculating the page rank is recursive but it may be computed by starting with any set of ranks and iterating the computation until it converges. Figure 3.2 demonstrates the propagation of rank from one pair of pages to another. Figure 3.3 shows a consistent steady state solution for a set of pages.

PageRank can represent a collaborative notion of authority or trust. For example, a user might prefer a news story simply because it is linked directly from the New York Times home page. Of course, such a story will receive quite a high PageRank simply because it is mentioned by a very important page. This seems to capture a kind of collaborative trust, as a page that is mentioned by a trustworthy or authoritative source, is more likely to be trustworthy or authoritative. Similarly, quality or importance seems to fit within this kind of circular definition.

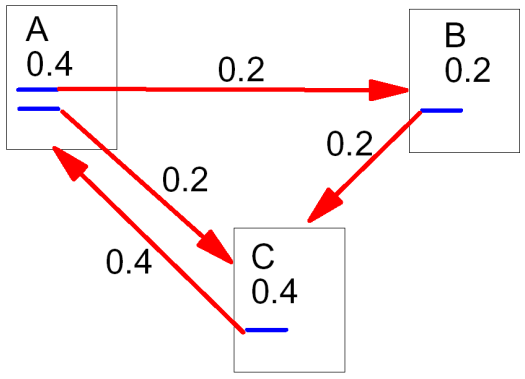


Figure 3.3: Simplified PageRank calculation

3.3 Internet statistics

Due to the fact that the media Internet is growing permanently with ever-increasing demand, it is getting more important to monitor the Web keeping the internet capacity and usage efficiency. This information is then used for technical, and economical decisions. There are a lot of projects examining the traffic, the structure, the technologies used, and much more. In this section I want to mention some of them with a brief description of their efforts.

The Internet Traffic Report [2] monitors the flow of data around the world. A 'ping' is used to measure round-trip travel time along major paths on the Internet. They have several servers in different areas of the globe, which perform the same ping at the same time. Each test server then compares the current response to past responses from the same test to determine if the response was bad or good. The scores from all test servers are averaged together into a single index. The date, time, latency, and loss values are stored in an archive. This data can be used for several types of decisions about building and using the network.

The Cooperative Association for Internet Data Analysis (CAIDA) The CAIDA resident in the San Diego Supercomputing Center (SDSC), is an

[2] `www.internettrafficreport.com`

extension of the University of California at San Diego (UCSD), where it was founded in 1997. It collects, monitors, analyses, and visualizes several forms of Internet traffic data concerning network topology, workload characterization, performance, routing, and multicast behavior. They build up analytic environments in which various forms of Internet traffic and routing data can be acquired, and analyzed. These analyses serve various purposes, including research, policy, education, and visualization.

The Internet Engineering Task Force (IETF) [3] is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture. The actual technical work of the IETF is done in its working groups, which are organized by topic into several areas. There is, for example, the *IP Performance Metrics (IPPM)* working group, which develops a set of standard metrics that can be applied to the quality, performance, and reliability of Internet data delivery services. Some of the metrics are connectivity, delay and loss, packet reordering, and bandwidth capacity, to mention just a few. These metrics will be designed such that they can be performed by network operators, end users, or independent testing groups.

The Internet Traffic Archive is a moderated repository to support widespread access to traces of Internet network traffic. This archive consists of different traces. One of them, for example, consists of all the requests made to the 1998 World Cup Web site between April 30, 1998 and July 26, 1998. During this period of time the site received 1 352.804.107 requests. One request stored consists of the timestamp of the request, requested URL, the number of bytes in the response, the method contained in the client's request, status of the request, type of the file requested, and the server indicating, which Web server handled the request. To ensure the privacy of each individual that visited the World Cup site, all of the client IP addresses have been removed and replaced with a unique integer identifier. The traces can be used to study net-

[3] `http://www.ietf.org/`

work dynamics, usage characteristics, and growth patterns, as well as providing trace-driven simulations.

In the following section I want to provide some numbers about the Internet to illustrate the largeness and the alteration of the media Internet.

The numbers provided below in Table 3.1 are the numbers of hosts provided by the *The Internet Software Consortium (ISC)*[4], which is a none-profit corporation.

Date of evaluation	Number of hosts in Austria	Total number of hosts
July 1995	40 696	6.6 Mio
July 1996	71 090	12.9 Mio
July 1997	87 408	19.5 Mio
July 1998	132 202	36.7 Mio
July 1999	203 774	56.2 Mio
July 2000	349 625	93.1 Mio
July 2001	600 752	125.9 Mio
July 2002	720 587	162.1 Mio

Table 3.1: Total number of hosts compared to the Austrian hosts

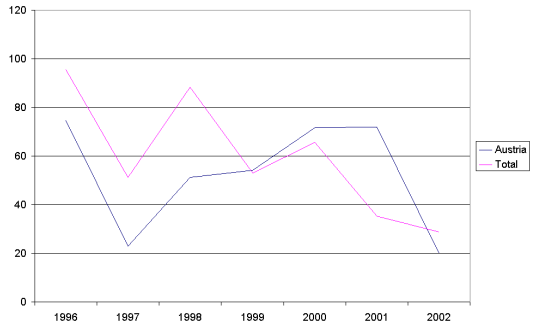


Figure 3.4: Growth rate of the hosts

In Figure 3.4 we see the growth rate of the hosts compared to the respective previous year. It can be seen that the number of hosts on the Internet has

[4] www.isc.org

nearly doubled in the time from July 1999 to July 2000. Whereas the growth rate of hosts in Austria was much less in this year. But in the period between July 2000 and July 2001 the hosts in Austria increased much more than the world average. This is an example of the anomaly in the Internet. Especially the growth rate of the Internet depends on a lot of different parameters and is not yet fully comprehensible.

Anyway, these numbers give us a first impression of the huge size of the hosts involved in the Internet.

3.4 Growing Interest in Web Analysis

With the popularity of the World Wide Web and the recognition of its worthiness of being archived we find numerous projects aiming at creating large-scale repositories containing excerpts and snapshots of Web data. Among the most famous of these we find, for example, the *Internet Archive* [41], located in the US, which, among many other collections, has the largest archive of Web pages from all over the world, donated by the search engine Alexa. The leading Web archiving project in Europe is the *Kulturaw3* project by the Swedish Royal National Library [2] described in Section 3.1. There are many more Web archives all over the world and their collections are growing rapidly.

The results of these analyses are in turn very important to provide input for solving problems coming up with the enormous growth of the Internet in the past years. Clever ranking algorithms allow search engines to provide much better results for the user queries.

Due to this growing potential of analyses there will be many efforts developing new algorithms, analyzing techniques, and interfaces for analyzing the WWW.

Chapter 4

Data Acquisition

In this chapter we take a look at the Web data, which is acquired by the AOLA project, and the additional data we use in this project. We defined different modules, which can be seen in Figure 4.1 to be able to separate this acquisition process. Section 4.1 outlines the data acquired by the AOLA program. The first module described in Section 4.2 called the *AOLA-module* contains the process of adjusting and transforming the data gathered by the AOLA project. The second module, which is called the *DNS-module*, comprises the processes of collecting additional data from the DNS[1] servers and is described in Section 4.3. The last module described in Section 4.4 is the *WHOIS-module* in which the process gathering additional data from the according WHOIS server is implemented.

4.1 Data acquired by the AOLA project

Web information is gathered actively using bulk collection, which is the acquisition of the material by Web-crawlers. Starting from a number of sites, they move to other sites following the links they find. Due to the highly interlinked structure of documents on the Internet, these robots are able to harvest autonomously a considerable portion of the Web. Yet, sites, which are not part of

[1]DNS is the acronym for Domain Name Service, which are the machines responsible for maintaining lists that translate Internet names to numbers and vice versa. DNS allows you to reference domain names instead of their actual IP address for easier recollection.

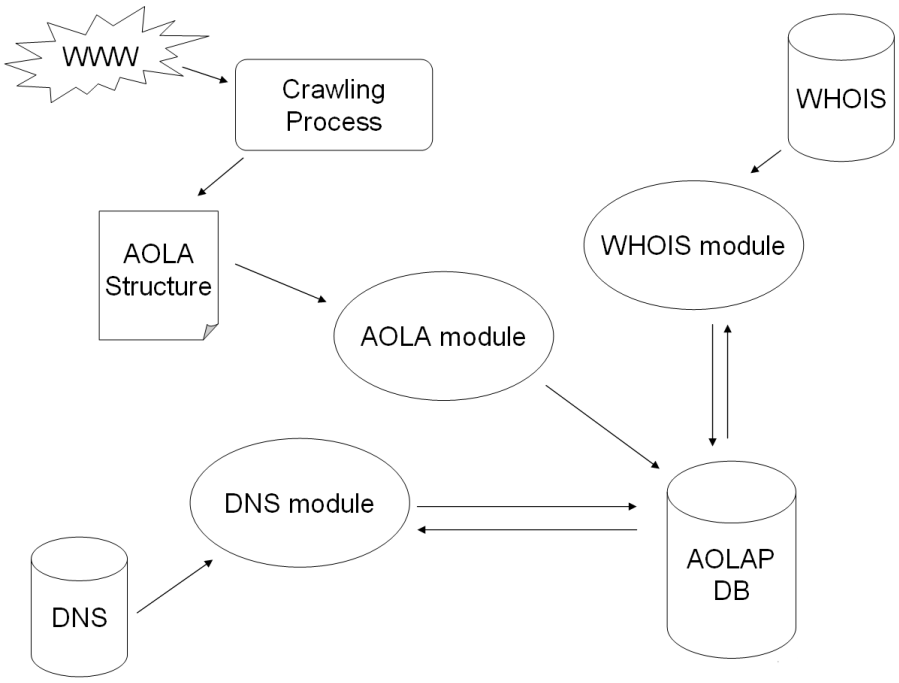


Figure 4.1: Data acquisition modules

the initial setting and are not linked by any other site, will not be collected. This part of the Internet, known as the deep Web, remains out of reach. Depending on the size of the given Web space, as well as the available bandwith, crawls may take several months for completion.

The AOLA archive is built up by actively collecting the data from the Web in contrast to the passive data acquisition where the archive is made up of documents submitted by the publishers.

For guaranteeing efficient and flexible operations on the data stored a well defined storage structure from the *Kulturarw3-project* [2] is used and can be seen in Figure 4.2.

Documents from the same Web server are grouped together into one directory. Because of the huge amount of Web servers, another partitioning layer is included not to overload the underlying operating system. To gain unique iden-

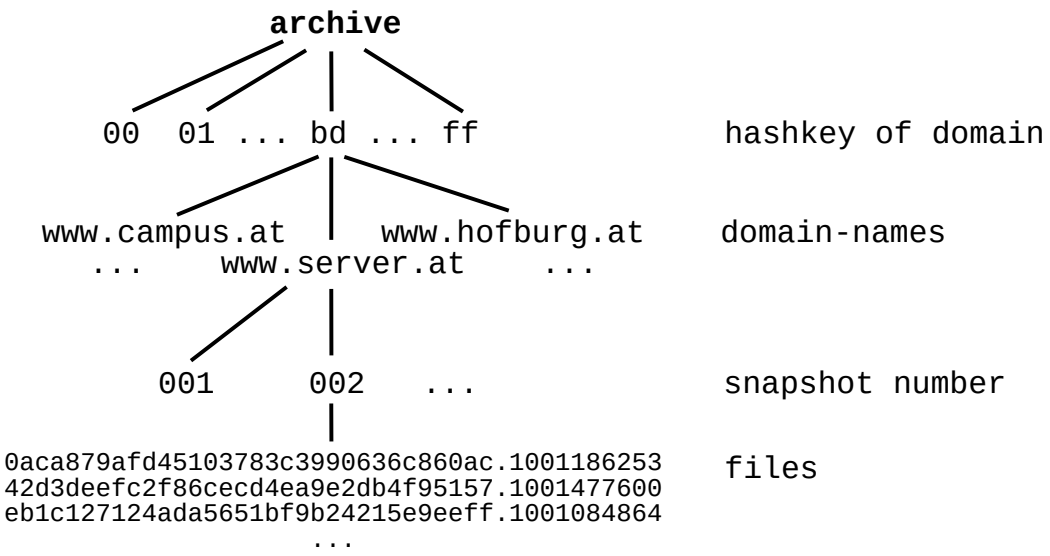


Figure 4.2: Scheme of the Kulturarw3 storage format

tifiers, a collision-proof checksum applying the RSA MD5 algorithm [2] is used that provides a unique sequence of 32 characters digesting any input. The first two characters of the MD5 checksum of the host name is used at the top level in order to split the servers as uniformly as possible into 256 different directories. Next, the server names are used as directory names. Following the IDs of already performed snapshots, a level further down the hierarchy, finally the files are to be found. The file name consists of a 32 characters long string, which is the MD5 checksum of the URL where the original data object was retrieved. Partitioned by a point, a time stamp is appended to this character string. Thus, a full path to a file retrieved from the Web server `www.server.at` could look, e.g., like this: *archive/bd/www.server.at/002/0aca879afd45103783c3990636c860ac.1001186253*.

All information about a document is stored in one individual file, encapsulated in MIME format. It has three separate parts as displayed in Figure 4.3. The first part contains the meta-data associated with the collection process, such as when it was collected. The second part contains the meta-data delivered by the Web server. This includes information provided as part of the http

[2]RSA Data Security,Inc. - MD5 Message-Digest Algorithm

protocol as well as other information provided by the server, such as the server software type and version, the operating system used by the server, date and time settings at the server, as well as last-modified dates for the respective file being downloaded. The actual content of the original file is to be found in the third part of the file.

We have to consider that it is impossible to get a correct and complete sample of the 'entire Austrian Web' due to the following characteristics of the Web data source:

- dynamic
- distributed and autonomous
- sites are sometimes down
- some people decide to not allow their sites to be indexed

Despite all this, I believe we have a reasonable representation of the actual structure of publicly accessible Austrian Web.

In the following sections I will describe the modules, implemented for gathering, adjusting, and transforming data used in our project.

4.2 AOLA-module

As the primary source of information, we use the data gathered by the *Austrian On-Line Archive (AOLA)* project. This module is implemented to adjust and transform the AOLA-data into a clear form, which will be used by the feeding process. This is done by a perl script. The files, which are stored in the AOLA archive have to be parsed and the useful information for our project is extracted into four different files. The third part of the AOLA-files is not parsed if it represents a file type out of a specified set like e.g. jpg, mp3, mov, etc. because in these files there are no links to be parsed.

Because of the large fraction of Web pages, having incorrect HTML, it was very difficult to design the parser. Of course, it is not possible to consider every

```
MIME-version: 1.0
Content-Type: multipart/mixed; boundary=aola_f2411dd811c7ab1187036b392c85e8df
HTTP-part: Archive-Info
HTTP-www-archiver: aola
HTTP-archiver-version: 0.01
HTTP-URL: http://www.ifs.tuwien.ac.at/~aola/
HTTP-Content-MD5: f2411dd811c7ab1187036b392c85e8df
HTTP-archive-time: 1001411155

                  --aola_f2411dd811c7ab1187036b392c85e8df

Content-Type: text/plain; charset="US-ascii"
HTTP-part: Header-Info

Connection: close
Date: Tue, 25 Sep 2001 10:51:08 GMT
Accept-Ranges: bytes
Server: Apache/1.3.12 (Unix)  (Red Hat/Linux) PHP/4.0.4-dev
Content-Length: 6542
Content-Type: text/html
ETag: "25dc04-2cc-3699b6aa"
Last-Modified: Mon, 04 Sep 2001 08:30:34 GMT
Client-Date: Tue, 25 Sep 2001 09:45:55 GMT
Client-Peer: 128.131.167.10:80
Content-Base: http://www.ifs.tuwien.ac.at/~aola/

                  --aola_f2411dd811c7ab1187036b392c85e8df

Content-Type: text/html
HTTP-part: Content

<html>
<head>
<title>AOLA - Austrian On-Line Archive</title>
<meta http-equiv="Content-Type" content="text/html">
<meta name="keywords" content="aola, austria, online, archive, digital, media
<script language="JavaScript">

<!--
function preloadImages(){
  var d=document;
  ...
The amount of information published on the Internet continues to grow at a tremendous rate.
Yet, contrary to conventional publications, little of what is published on the World Wide Web
is actually preserved in an archive.
The need for creating an archive of the information published on the Web, being part of
humankind's cultural heritage, is being recognised by national libraries worldwide, and
resulted in the creation of numerous projects addressing these issues.
  ...
</body> </html>

                  --aola_f2411dd811c7ab1187036b392c85e8df--
```

Figure 4.3: Structure of an archived file
(`http://www.ifs.tuwien.ac.at/~aola`)

Entry	Example
host	www.tuwien.ac.at
URL	http://www.tuwien.ac.at/forschung/ nachrichten/a-lola.htm
size	5385
MIME type and character set	text/html charset=iso-8859-1
last change client date	Mon, 10 Apr 2000 14:39:38 GMT
internal links	2
external links	1
mail links	0
timestamp	1001483086

Table 4.1: Examples of the entries in the file *stats.data*

HTML object on the pages, because a lot of them are simply wrong and hence not working.

In the following section I will describe the output files of this module.

- *stats.data*
  This file contains the host, URL, amount of internal, and external links, whereby the number of internal links arises from all links pointing to the same domain as of the host. All other links are denoted as 'external links'. Mail links are the number of e-mail addresses of the specific page. The host can be extracted from the URL when the data is loaded into the database. The size of the specific page is stated in kilobyte. Also the MIME type including the character set of each page is stored in this file. As already said before, we can not guarantee the accuracy of the information like the MIME type or the 'last change client date', gathered from the Web servers. Sample entries of this file can be seen in Table 4.1.

- *forms.data*
  In this file the number of forms on a specific page and the proper number of interactive items for each form are stored. Sample entries of this file can be seen in Table 4.2.

Entry	Example
host	www.tuwien.ac.at
URL	http://www.tuwien.ac.at/forschung/ nachrichten/a-lola.htm
number of forms	2
no. of interactive items of form 1	18
no. of interactive items of form 2	7

Table 4.2: Examples of the entries in the file *forms.data*

Entry	Example
host	www.tuwien.ac.at
link host	www.univie.ac.at
URL	http://www.tuwien.ac.at/forschung/ nachrichten/a-lola.htm
number of links	2
total number of external links	9

Table 4.3: Examples of the entries in the file *links.data*

- *links.data*

  This file contains for each page the host where each link points to. Additionally the number of the links to the specific host as well as the total number of external links from the specific page are stored in this file. The protocol used by the link will be extracted from the link URL. In a later version of this module we changed the file *links.data* to *AllLinks.data* where not only the links to a specific domain is countered, but to every specific page of the domain linked to. To feed this data into the database, which needs much more space and much more time, will be due to time constrains, a future work. Sample entries of this file can be seen in Table 4.3.

- *server.data*

Entry	Example
host	www.atv.tuwien.ac.at
Web server	Apache/1.3.12
operating system	(Unix)
modules	PHP/4.0.4pl1 tomcat/1.0
port	80

Table 4.4: Examples of the entries in the file *server.data*

This file contains the information of the server for each host. The type, and the version of the Web server as well as the type and version of the underlying operating system are stored. Additionally the modules and versions of them, which are installed on each specific host, are stored in this file.

4.3 DNS-module

This module is implemented to enrich the existing data with other useful data gathered from the DNS (Domain Name Server) server responsible for the '.at' domain.

The DNS server is looked up with perl functions like *inet_ntoa()* to get the IP addresses of the hosts. The input file of this script is the *server.data* file from which the script gets the host as input. The output file is called *LookupData.data* and it contains beside the host the information if the host was available at a specific time, the IP address of the host (up to four different IPs can be stored) and the aliases (up to five different). An alias is one of a set of domain names of one Internet resource. A lot of Web servers support hosting multiple domains on one server. The different sites will have one common IP address but different domain names. There is one primary domain set at the Web server as the default Web site. The domain name of each IP is looked up and compared with the domain name of the crawled host. If the requested domain name does not match with the crawled domain name, it is additionally stored as the primary domain

name of this host.

In this module the reachability of each host is checked too. Therefore the host is pinged. Because a lot of hosts are just temporarily not available we save the time when the ping is done as well.

When the module is used in other projects with other kinds of data, you just have to fit the interface, which is in this case just a file of the hosts we want to enrich. You do not have to worry about things like separators, because it is very easy to adapt these scripts in order to get the ability to achieve different requirements. And, of course, due to lookup the DNS server over the Internet a stable connection to the net is necessary. We looked up more than 1600 hosts per hour, which means that in our case the scripts runs about 3 days. For sure this time measures depends from the bandwith of the Internet connection and it is heavily dependent from the performance of the different DNS servers.

4.4 WHOIS-module

This module gathers data from the specific WHOIS server. The WHOIS database contains information about IP networks, domain names, their administrators and other technical info. There are a lot of different WHOIS servers, which are responsible for different domains. The WHOIS server is completely separate from DNS servers. Because of time constraints, we decided to lookup just the hosts in the '.at' domain, which includes most of the total number of hosts. The responsible WHOIS server for the domain '.at' is the ripe[3] server. A sample result from the query `tuwien.ac.at` can be seen in Figure 4.4.

The result set can contain different objects. In this case there are a so called 'domainobject', and two 'person' objects. The first column represents the attribute, and the second column specifies the value of the attribute. Very often, there are further descriptions in the second column, as in this result set, for example, the '[organization]'. Beneath the attribute 'descr' (description) there are contact information like name, address, and telephone number of the registered organization. Further attributes of the 'domainobject' are:

[3] `www.ripe.net`

- *admin-c*: References an on-site administrative contact
- *tech-c*: References a technical contact
- *zone-c*: References a zone contact where a DNS server is responsible for a specific zone of the Internet
- *nserver*: Specifies the nameservers of the domain
- *remarks*: Can be any information entered, in this case it is the IP address of the nameserver
- *changed*: Specifies when the object was updated
- *source*: Specifies the registry where the object is registered

The second and third part of the example are so called 'person objects', which contain some address and contact information about the persons registered.

Further objects gathered are, for example, the 'Inetnum objects' received from the WHOIS database for each host. An 'Inetnum object' contains information on allocations and assignments of the IPv4 address space.

- *netname*
  The name of the specific netblock
- *netblock*
  This is the range of the IP addresses of the specific netblock
- *maintainers*
  The owner of the netblock in which the specific host is addressed

We query the hosts from our database and send the request for each host to the according WHOIS server. The response information from the WHOIS server is stored directly into our database. We store the organization or, if not available, the name of the first contact person into the table *owner*. The zip code gathered is stored into the table *address*. On the basis of this zip code we lookup the city and the county from an external file where all the zip codes and

File format	Finnish	Swedish
HTML	40%	50%
GIF	25%	19%
JPEG	20%	23%
PDF	3%	1%

Table 4.5: Comparison of Finnish and Swedish most popular file formats

their respective cities are stored. One problem we encountered was that some WHOIS servers including the RIPE server have request limits per day. This means that we could not request more than 600 hosts per day.

4.5 Summary

The archive currently consists of approximately 13 Mio URLs from more than 133.400 individual Web sites, which amount to about 184.000 different sites including alias names of servers. The Swedish Kulturaw3 project has gathered in the last run crawled in the year 2001, about 30 Mio unique URLs.

Also the Finnish Web archive consists 42 million different URLs. The most popular file formats of the Finnish and Swedish archive are presented in Table 4.5.

As you can see in Table 4.5, the most file formats used are quite similar in the Swedish Web space compared to the Finnish Web space.

In this chapter I showed the structure of the data gathered by the AOLA program from the Web. Furthermore, I explained the four different modules implemented to transform the data gathered from the Web, and to add additional useful data from different sources. The *AOLA-module*, used to transform the Web data to a useful and simple structure for our project, is shown in Section 4.2. In Section 4.3 the *DNS-module* to gather the data from the DNS servers is explained. The *WHOIS-module*, used for the lookups at the WHOIS servers, were shown in Section 4.4.

```
domain:    tuwien.ac.at
descr:     [organization]:Vienna University of Technology Communication Group
descr:     [street address]:Wiedner Hauptstrasse 8-10/020
descr:     [postal code]:A-1040
descr:     [city]:Wien
descr:     [country]:Austria
descr:     [phone]:+43 1 58801 42040
descr:     [fax-no]:+43 1 58801 42099
descr:     [e-mail]:hostmaster@noc.tuwien.ac.at
admin-c:   JD766159-NICAT
tech-c:    JK561471-NICAT
zone-c:    JK561471-NICAT
nserver:   tunamec.tuwien.ac.at
remarks:   192.35.241.70
nserver:   tunamed.tuwien.ac.at
remarks:   192.35.241.71
changed:   20010309 14:39:40
source:    AT-DOM

person:    Johannes Demel
address:   ZID der TU Wien
address:   Wiedner Hauptstrasse 8-10/020
address:   A-1040 Wien
address:   AUSTRIA
phone:     +43 1 58801 42040
fax-no:    +43 1 58801 42099
e-mail:    demel@zid.tuwien.ac.at
nic-hdl:   JD766159-NICAT
changed:   20010104 13:52:25
source:    AT-DOM

person:    Johann Kainrath
address:   University of Technology
address:    Vienna
address:    Computing Services
address:   Wiedner Hauptstrasse 8-10/020
address:   A-1040 Wien
address:   Austria
phone:     +43 1 58801 42045
fax-no:    +43 1 58801 42099
e-mail:    kainrath@noc.tuwien.ac.at
nic-hdl:   JK561471-NICAT
changed:   20000825 14:15:52
source:    AT-DOM
```

Figure 4.4: Result from the query 'tuwien.ac.at' at the RIPE WHOIS server

Chapter 5

Database Design

One of the most tricky tasks was the design of the database model finding the tradeoff between designing it completely denormalized and having too much redundancy. The DB schema has grown to quite a complex structure of linked tables. I will describe each table and the most important columns of each below.

The current model can be divided into two parts. The first part arises from all tables containing data referring to the Austrian Web hosts, described in Section 5.1 and 5.2. The second part consists of tables containing data about the hosts where the links of the Austrian Web point to, described in Section 5.4. The table connecting these parts represents the links stored, described in Section 5.3. This table forms the central fact table in the Data Warehouse. The whole but simplified scheme of the model is shown in Figure 5.1. A lot more tables are involved in our project. For reasons of clearness, I added just the most important ones.

At this point I want to define the commonly used expressions *Web site*, *Web host*, *domain*, and *Web page*. A Web site is an organized collection of Web pages on a specific topic maintained by a single person or group. Several Web sites can be located on one Web host. Web sites in turn, contain several more or less structured Web pages, which contain the actual data. A Web site is not equal to a domain. Thousands of sites can be hosted on one domain (e.g. the domain `www.geocities.com`). Due to the fact that we do not have content analysis yet, we do not distinguish between different Web sites on one Web host. This means

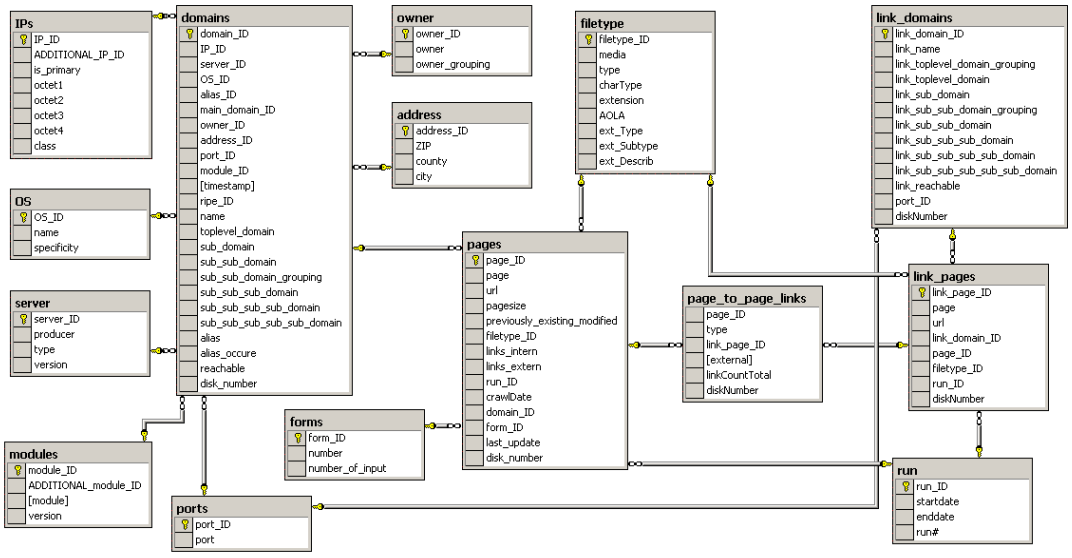


Figure 5.1: DB model

that in our model the Web host and the Web site can be seen as the same and are stored in the table *domains*.

To keep track of the database model I will describe different parts of the database separately. The table *domains* with the surrounding tables where the properties of the hosts like the IPs, operating system, etc. are stored, is shown in Section 5.1. In Section 5.2 the pages of the domains and their respective data is described in detail. The links of the pages, stored in the table *page_to_page_links*, are described in Section 5.3. The data of linked 'foreign' pages and domains, stored in the tables *link_pages*, and *link_domains*, are described in Section 5.4.

5.1 Domains and their respective data

The part of the DB, containing the data of the Austrian Web sites is shown in Figure 5.2. As we can see it is a classical *Snowflake schema*. The data is maintained in a single fact table (*domains*) at the center with additional dimension data stored in dimension tables. Each dimension table is directly

related to and joined to the fact table by a key column.

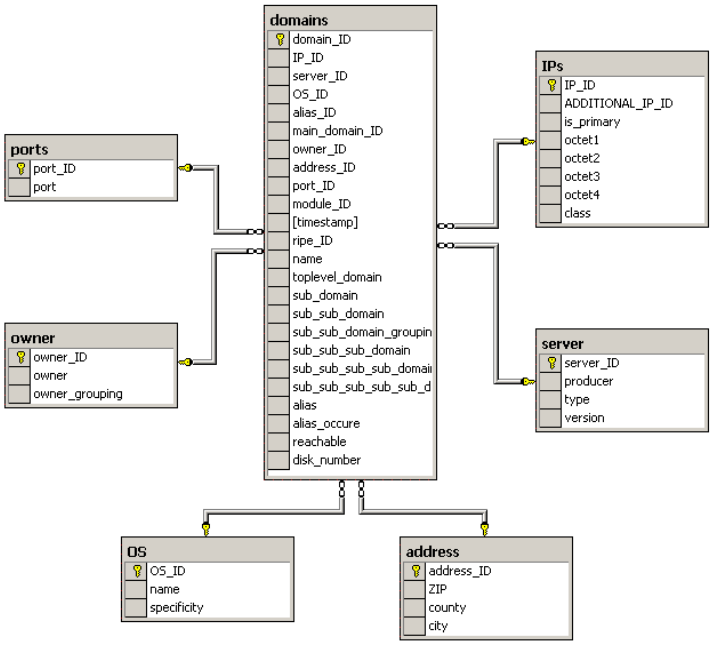


Figure 5.2: Domains and dimension data

Below I provide a brief description of the tables.

- *domains:* This table contains the names of the Web hosts organized by sub-level domains. For example, the host `www.tuwien.ac.at` of the Vienna University of Technology is split into four columns. The top-level domain `'at'` is stored in the column *toplevel_domain*, the sub-domain `'ac'` in the column *sub_domain*, the `'tuwien'` in the column *sub_sub_sub_domain*, and the `'www'` is stored in the column *sub_sub_sub_sub_domain*. The sub-level domain `'ac'` is a big network just for educational establishment. In the other columns regarding the domain name there are just placeholders to preserve the connection of the whole domain name as you can see in Table 5.1. This allows us to drill down through the address space during analysis.

Column	Entry
toplevel_domain	at
sub_domain	ac
sub_sub_domain	tuwien
sub_sub_sub_domain	www
sub_sub_sub_sub_domain	#NZ
sub_sub_sub_sub_sub_domain	#NZ

Table 5.1: Example of the domain www.tuwien.ac.at stored in the database (#NZ is a placeholder)

In the next example I will show a problem we were confronted with. Let us assume we have to store the host `www.orf.at`. At the first glance we would think of saving the `'at'` into the column *toplevel_domain*, the `'orf'` into *sub_domain*, and the `'www'` in the column *sub_sub_domain*. But if we aggregate this dimension in the Data Warehouse, we would have the `'www'` from the `'www.orf.at'` and the `'tuwien'` at the same level. But actually, we want to have both `'www'` at the same level and the `'orf'` at the same level of `'tuwien'`. To solve this problem, we stored the `'orf'` in the column *sub_sub_domain* and just put a placeholder in the column *sub_domain*, which can be seen in Table 5.2. The field for the sub-level domain thus is reserved for the major subdomains in the Austrian Web space, i.e. '.ac', '.co', '.gv', denoting the academic, commercial, and governmental subdomains, respectively.

Next, I provide a short description of the columns of this table.

- *domain_ID*:
  Key column.
- *IP_ID*:
  Foreign key of table *IPs* where the according IP address is stored.
- *server_ID*:
  Foreign key of table *server* where the Web server used is stored.

Column	Entry
toplevel_domain	at
sub_domain	#NZ
sub_sub_domain	orf
sub_sub_sub_domain	www
sub_sub_sub_sub_domain	#NZ
sub_sub_sub_sub_sub_domain	#NZ

Table 5.2: Example of the domain www.orf.at stored in the database (#NZ is a placeholder)

- *OS_ID*:
  Foreign key of table *OS* where the underlying operating system is stored.
- *alias_ID*:
  Key of the alias entry of the Web host. As described in Section 6.1 the according aliases of the host are stored in table *domains* as well.
- *main_domain_ID*:
  A lot of Web servers support hosting multiple domains on one server. The different sites will have one common IP address, but different domain names. There is one primary domain set at the Web server as the default Web site. This site will be displayed by entering the IP address into your browser. If there are multiple domains set, the primary domain is stored in this column.
- *owner_ID*:
  Foreign key of table *owner* where the registered owners of the Web sites are stored.
- *address_ID*:
  Foreign key of table *address* where the addresses registered at the respective WHOIS servers are stored.
- *port_ID*:

Foreign key of table *ports* where the according ports are stored.

- *module_ID*:
  Foreign key of table *modules* where the modules installed on the Web servers are stored.
- *timestamp*:
  Foreign key of table *time*. Represents the timestamp of the date the host is pinged and the DNS server is contacted.
- *name*:
  Whole name of the Web hosts.
- *toplevel_domain*:
  Top-level domain of the host.
- *sub_domain*:
  Sub-level domain of the host. If the host does not have a well known sub-level domain like `'ac'`, `'gv'`, etc. the value `'#NZ'` will be entered in this column.
- *sub_sub_domain*:
  Sub-sub-level domain of the host.
- *sub_sub_domain_grouping*:
  This column is filled with the first character of the sub-sub-level domain to group the entries alphabetically. This column is used in the Data Warehouse for a grouping level of the dimension. If the first symbol is not a character, the symbol `'#'` is entered instead.
- *sub_sub_sub_domain*:
  Sub-sub-sub-level domain of the host. If the host entry has no sub-sub-sub-level domain as e.g. `univie.ac.at`, this column and all following sub-level-domains are filled with the placeholder `'#NZ'`.
- *sub_sub_sub_sub_domain*:
  Fourth sub-level domain of the host.
- *sub_sub_sub_sub_sub_domain*:
  Fifth sub-level domain of the host.

- *alias*:
  The first alias of the host.
- *alias_occure*:
  This column represents the number of aliases of the specific host.
- *reachable*:
  Flag if the domain is reachable at a specific time stored in the column *timestamp*.
- *whois*:
  Flag for internal use during the feeding process. It shows if the WHOIS server of this host was already contacted or not.

Identical hosts, i.e. hosts having the same IP address, yet reachable via different domain names, are stored multiple times in this table to reflect the actual domain name space.

The table *domain* is not a critical table of the database, as the number of Austrian Web hosts keeps within reasonable limits. At the moment we count 184.022 entries in the table *domains* compared to about 13 Mio entries in the table *pages*. In terms of growth of this table it would be no problem to enhance the Web data fed into the database.

- *IPs:* In case the host has a valid DNS entry, the IP addresses of the Web hosts, separated into the octets ranging from 0 to 255, are stored in this table. This is used to identify the different types of IP nets, i.e. class A, B and C networks. Class A addresses are reserved for very large networks such as the ARPANET and other national wide area networks. Class B addresses are allocated to organizations that operate networks likely to contain more than 255 computers and Class C addresses are allocated to all other network operators. IP addresses of servers reachable via multiple domain names but having the same IP address are stored only once. Beside the primary interface's IP address of a host, there may be more IP addresses the host responds to. In this case we store the ID of the next IP address in the column *ADDITIONAL_IP_ID*. And the ID of the

further IP address is again stored in this column of the additional IP address record. To mark the primary IP address, we set the flag to `1` in the column *is_primary*, otherwise it would be `0`. There are 154.735 entries stored in this table, which is a bit less then the number of entries of the table *domains*. In spite of the fact that there is sometimes more then one IP address for one host stored, this number is not surprising because of two points. First, we have to consider that the aliases of the hosts, which have the same IP address, are stored in the table *domains* too. Second, we could not gather the IP address of all domains stored in the database.

- *server:* In this table server information like the type and version of the server (e.g. MS IIS, Vers 5.0) is stored, structured hierarchically by producer, product, and version. Please note that no checking of the validity of this information is performed during the download, i.e. disguised servers are not identified as such. Currently there are about 105 servers in a total of 478 versions by 42 producers stored in this table.

- *modules:* This table represents the data of the add-on modules installed on the Web servers. There are the columns *module* and *version* where the name and the version of the module is stored. Due to the fact that on one Web server there are normally several add-on modules installed, there is a 1-to-n relation between the hosts stored in the table *domains* and the modules. To handle this, we added a column namely *ADDITIONAL_module_ID* where the ID of the next module of the same host is stored. If there are more then two modules installed at one host, the next module again contains an additional module-ID, where the ID of the third module is stored, and so on.

- *OS:* The table *OS* contains the reported name and the specificity of the operating system of the hosts. Again, no checking of the validity of this information is performed.

- *owner, and address:* These two tables are filled with data from the WHOIS server. In the first one the owner of the Web host, and in the second one, the address registered at the WHOIS server, is stored in a hierarchical

structure starting from the ZIP code over the city to the county of Austria. The table *address* contains about 1200 entries including more then 900 different townships of Austria.

- *ports:* The TCP ports over which the hosts were accessed, are stored in this table. A link can contain the host like e.g. in the URL `www.tuwien.ac.at:80`. When we have found such a port in a link parsed, we also stored it into this table. Otherwise we stored the default value of the protocol. The default ports accessing a host via the 'http' protocol is `80`. Each protocol uses a special port, but due to various reasons the default values are sometimes changed. In this case the ports must be added to the URL. To read more about ports please refer to the IANA (Internet Assigned Numbers Authority)[1].

5.2 Pages and their respective data

In Figure 5.3 the table *pages* with its according dimension tables is shown. The schema is as the data schema of the table *domains* described above a *Snowflake schema* where the tables *filetype, forms, time, and run* are used as the according dimension tables.

Below I provide a description of the fact table used in this cube.

- *pages:* In this table all pages gathered from the AOLA database are stored. There is a column *page* were the name of the page is stored, a column *url* containing the URL of the specific page. Further information includes the size of the page, crawl date, as well as the date of the last modification for the downloaded page, if provided by the server. Additionally, the number of internal and external links are stored in this table. Further entries are the key entries for the surrounding dimension tables like *filetypes*, *forms*, etc.

There is one page entry for each unique URL of a domain in this table. In other words there are several entries of the same page (e.g. start.asp) of the same

[1] http://www.iana.org/assignments/port-numbers

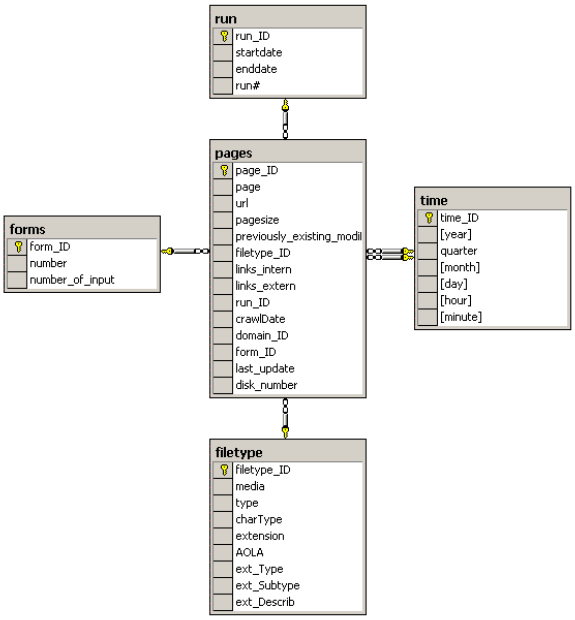


Figure 5.3: Pages and dimension data

domain but with a different URL like `http://www.tuwien.ac.at/forschung/start.asp?enter=1` and `http://www.tuwien.ac.at/forschung/start.asp?enter=2`. This table is used as a fact table surrounded by the dimensions *filetype*, *forms*, etc. as well as a dimension table for the fact table *domains*, which is described in Chapter 7. In terms of growth of table *pages*, it is a very critical one. At the moment we count 13.063.494 entries stored in this table. Due to the fact that each URL gathered from the Web represents a new entry in this table, the growth rate is very high. Thus, it is very important to take some measures to improve the performance of processes using this table, e.g. by tuning the table by setting indexes.

Following a brief description of the dimension tables.

- *forms:* This table stores, the number of forms per occurring Austrian page and the total amount of fields of a specific page to facilitate analysis of interactive Web pages, types and amount of interaction encountered, etc. The table is filled with 200 different number entries, which means

Column	Entry
page	lehrsuchhilfe.html
url	http://www.tuwien.ac.at/histu/ hilfe/lehrsuchhilfe.html
pagesize	2571
links_intern	7
links_extern	0

Table 5.3: Example of the page 'lehrsuchhilfe.html' stored in table *pages*

Column	Description
form_ID	key column
number	number of forms per page
number_of_input	number of input fields

Table 5.4: Table *forms*

that it is possible to save the number of form per page if the page has less then 200 forms. Additionally there are 200 different entries for the columns *number_of_input* for each form. That means that we can store the number of input fields of a form if it does not exceed 200 fields. Based on our experience, we can nearly eliminate the appearance of more forms or input fields. If we will find pages with more forms or input fields we will join this page to the entry '200+'. This entails just a minimal distortion of the results which is negligible.

- *filetype:* This table contains the different file types of the pages in the archive, as well as those of the foreign pages. The information is structured hierarchically by *media* (e.g., application, video, or image), followed by the *MIME type* and the *filename extension*. The character set used is stored as additional info. As basis for this structuring, both the MIME type provided with the downloaded page, as well as the file extension are used,

Column	Description
filetype_ID	key column
media	text
type	html
charType	charset=windows-1252
extension	html
AOLA	1
ext_Type	text
ext_Subtype	html
ext_Describ	html

Table 5.5: Table *filetype*

forming two independent dimensions. This separation is necessary due to the fact that the information provided both by the MIME type, as well as by the file extensions is prone to errors, and quite frequently these two dimensions do not correspond to each other. Retaining both types of information domains thus provides greater flexibility in the analysis. In Table 5.5 I will provide a sample table entry.

In the column *charType* the character type gathered from the Web server is stored. The column *AOLA* is a flag to show whether the according page of this filetype is an Austrian page (AOLA = 1) or not (AOLA = 0). The data based on the extension is stored in the columns with the prefix 'ext'. This data is entered manually. We compared the extension of the files with the official list of all MIME types assigned and listed by the IANA((Internet Assigned Numbers Authority) [44].

- *run:* In order to be able to compare the characteristics of the Austrian Web over time, we have to compare data from different crawls. For each snapshot we define a run number, start and end date, stored in table *run*. As in most other analyses, in time series analysis it is assumed that the data consist of a systematic pattern and random noise (error), which usually makes the pattern difficult to identify. The goal of time series

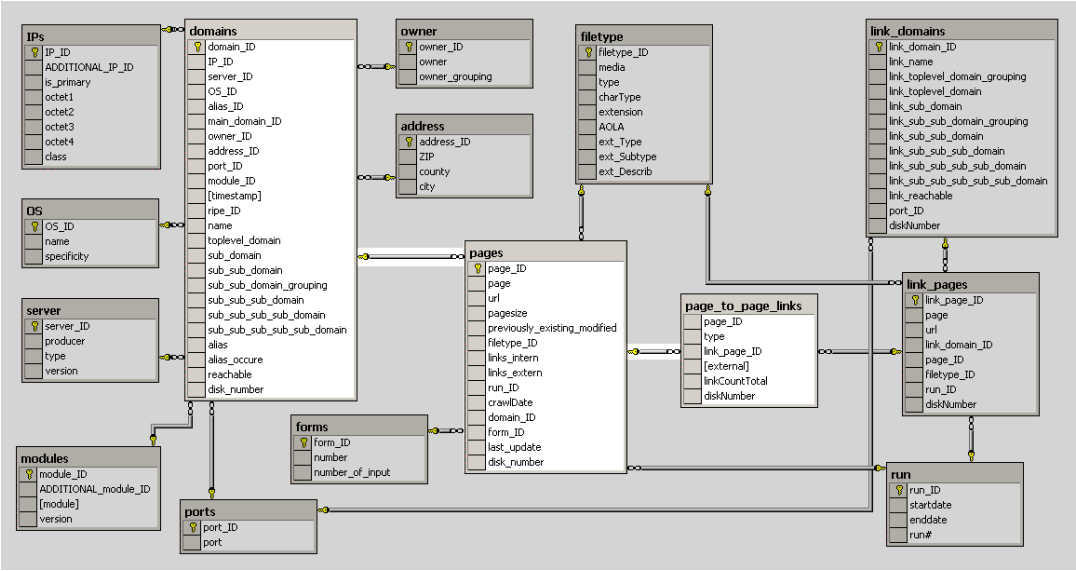


Figure 5.4: Domains - Pages - Links axis

analysis techniques is to find some patterns represented by the sequence of observations. When using Web data for time series analysis it is very important to choose the crawling dates very carefully. Think of Web sites, which are during the month quite unchanged and change completely at the beginning of each month like e.g. monthly news portals.

5.3 Domains - Pages - Links

In this section I will describe the axis *Domains - Pages* and their respective links as it can be seen in Figure 5.4.

In the following paragraph I will describe the table *page_to_page_links*. The description of the table *domains* can be found in the Section 5.1 and of the table *pages* in Section 5.2.

- *page_to_page_links:* All the links we gathered are stored in this table. In the column *type* there are the different prefixes of the URL, which indicate the protocol (http, https, ftp, etc.) of the stored link. *External* is an

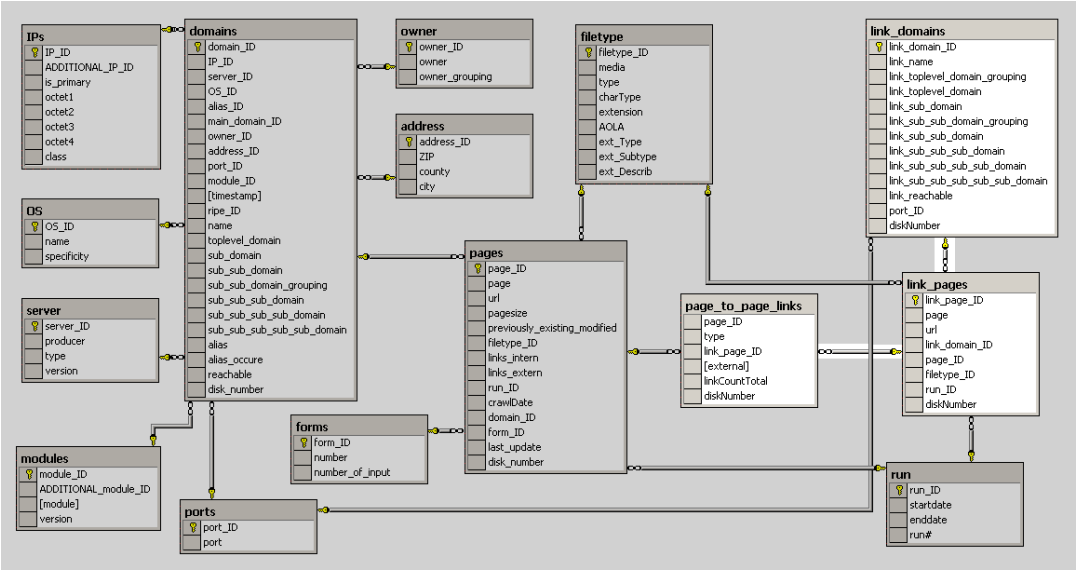


Figure 5.5: Links - link_Pages - link_Domains axis

additional column, which is used to differentiate between external and internal links, i.e., if the link references a page from another Web host or from the same domain. This table is the connection between the Austrian Web sites and the external Web sites. It is joined between the table *pages* and *link_pages*. In the Data Warehouse this table will be the fact table.

5.4 Links - linked Pages - linked Domains

This section describes the similar axis of the database but it contains the data from the external Web sites. The tables and their relations can be seen in Figure 5.5. These tables are called *link tables*, because the data, fed into these tables, represents the information where the specific Web site links to.

Because it is out of our focus, there are not so many dimension tables at this axis. For example, we did not query the WHOIS database or the DNS server for additional data about the entries of the tables where the linked pages and domains are stored. Hence the link tables have less columns then the tables

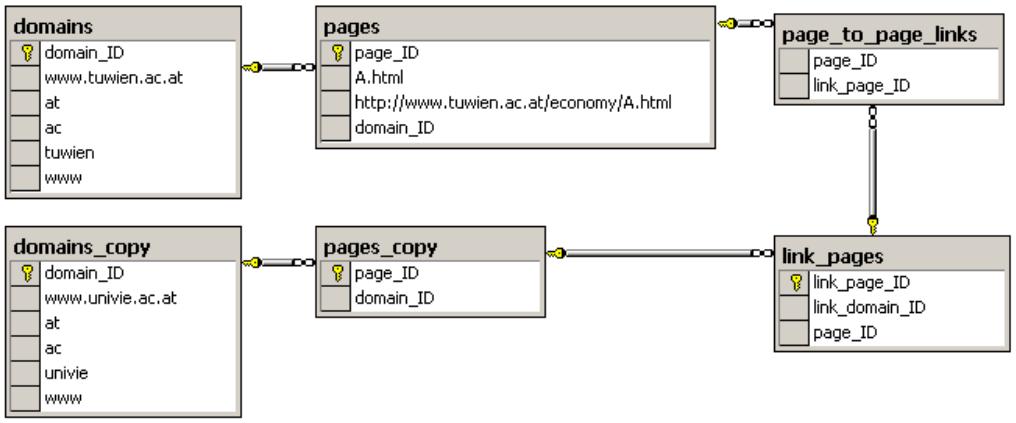


Figure 5.6: Example of a link stored in the DB

where the Austrian Web data is stored. Because the OLAP tool we used (described in Section 8.1) does not allow loops in the data schemes we decided to copy the tables *pages* and *domains*. A link of an Austrian Web site pointing to another Austrian Web site is stored in the table *page_to_page_links* joined with the table *pages_copy*. I provide the following example to illustrate this scenario: A page 'A.html' on the Web site `www.tuwien.ac.at`, with the URL `http://www.tuwien.ac.at/economy/A.html` contains a link to a page on the site `www.univie.ac.at`. The entries in the respective tables can be seen in Figure 5.6.

- *page_to_page_links:* As described above this table contains all the links gathered from the Austrian Web space.
- *link_pages:* This table contains all the so-called foreign link pages, i.e., pages referenced by pages of the AOLA database, which themselves are not in the Austrian Web space, and thus not part of the AOLA archive.

  In Table 5.6 you can see the columns of the table including a brief description.
- *link_domains:* This table stores the Web hosts, which are not in the Austrian Web space (so-called foreign Web hosts) but are linked to by the

Column	Description
link_page_ID	key column
link_domain_ID	second key column
page_ID	ID to join this table with the table *pages_copy*
domain_ID	second ID to join this table with the table *pages_copy*
filetype_ID	ID to join with the table *filetype*
run_ID	ID to join with the table *run*
page	page name (e.g. index.html)
url	URL (e.g. `http://www.tuwien.ac.at/home/index.html`)

Table 5.6: Example of the page 'lehrsuchhilfe.html' stored in the table *link_pages*

Austrian hosts. The sub-level domains are stored separately as in the table *domains* described above. If the link of the Web site consists of just an IP address, we looked up the host name of the IP address. If the host was not reachable, we sat the flag in the column *reachable* to zero.

5.5 Conclusion

As shown before the database schema has become quite complex. During the design phase the schema had to be adjusted several times. When we started testing the model by filling some sample data, and further started using the analyzing tool, we noticed that we had to make some further changes to the database model. Not all dimensions could handle the joins of the data tables we created. For example, we had to copy the tables *domains* and *pages* to reproduce the links starting from an Austrian page pointing to another Austrian page archived. Further we created some columns for internal use e.g. to keep a good overview of the data loaded into the database, we separated the entries into six different loading processes.

In this chapter I described the database tables as well as their connection among themselves. The data about the Austrian Web hosts stored in the tables *domains*, *IPs*, *server*, *OS*, *owner*, *address*, and *ports* was described in Section 5.1.

In Section 5.2 I provided an overview of the Austrian Web pages and their respective data. Section 5.3 described the axis 'Domains - Pages - Links' of the data model. A detailed description of the linked Web sites and their according data as well as the connection to the Austrian Web sites was provided by the axis 'Links - linked Pages - linked Domains' described in Section 5.4.

Chapter 6

Feeding Process

Much time has been spent feeding the database with the data crawled. Because of the huge volumes involved, this simple task is not trivial. For example, more then 40 Mio links have to be filled into the table *page_to_page_links*.

In this chapter I will describe the module implemented to fill the database with the data gathered from different sources. For better understanding I have divided this process into seven different steps, Section 6.1, and 6.3 describe the feeding of the database with the data from the Austrian Web sites. Section 6.2 explains the process of filling the according tables with the data from the Web servers of the hosts and with the additional data gathered from the WHOIS servers. In Sections 6.4, 6.5, and 6.6 I will outline the process of feeding the respective data of the linked Web sites into the database. In Section 6.7 I will describe some manual and additional adjustments to the database.

6.1 Step 1: Filling the tables of the Austrian Web Sites

The base data of this step is the file *LookupData.data* where the host, according IP addresses, aliases, reachability of the host, and time of the DNS request are stored. A perl script parses the file and transforms and adjusts the data to store it into the database. I will show this with a sample data set of the file

LookupData.data.

```
eeasvr.ea.tuwien.ac.at 1 128.130.75.40 # # # #
www.ea.tuwien.ac.at # # # # # Fri Mar 8 09:11:17 2002
```

Table 6.1: Sample data set of the file *LookupData.data*

The data entries in the file *LookupData.data* shown in Table 6.1 are separated by a blank and the placeholder for null values are '#'. That indicates that the host name is 'eeasvr.ea.tuwien.ac.at', which is reachable (that shows us the '1' after the hostname, otherwise it would be zero) and the IP address of it is `128.130.75.40`. There is one alias name of the host named `www.ea.tuwien.ac.at` and all this data is gathered on the 8th of March 2002. The perl script splits the host name and stores it into the according columns of the table *domains*. The domain_ID of the first alias of the host is stored together with a number denoting the total amount of aliases associated to this host (in this case '1'). The flag that the domain was reachable is also stored in the table *domains*. The IP address is also split and stored in the according columns of the table *IPs*. Depending on the first octet of the IP address the class of the IP address is evaluated and stored in the column *class*(A, B, or C). The date has to be parsed and day, month, year and time have to be filtered out. It is important to account for the great variety of different time formats depending on the Web servers. Afterwards, we query the time table to store the right time_ID in the table *domains*. The alias is also stored into the table *domains* with the key stored in the row entry of the host. In other words two data records in the table *domains* were created. One for the host, and the second entry for the alias itself. To illustrate this example the stored data sets are shown in Figure 6.1.

	domain_ID	name	toplev	sub_d	sub_sub_	sub_su	sub_sub_s	IP_ID	alias_ID	alias	alias_occure	reachable
▶	46574	eeasvr.ea.tuwien.ac.at	at	ac	tuwien	ea	www	36204	0	www.ea.tuwien.ac.at	0	1
	46575	eeasvr.ea.tuwien.ac.at	at	ac	tuwien	ea	eeasvr	36204	46574	<NULL>	1	1
*												

Figure 6.1: Sample data set stored in the table *Domains*

As said above this table is not a critical table because the growth rate of this

table is not very high. Hence the loading process of this table is not very time critical too.

6.2 Step 2: Filling the Server and the WHOIS Data

In this step we fill the database with the server information, which we gathered from the various Web servers. The input file for this step is the file *server.data* where all the information about the underlying server of each host is stored. Table 6.2 shows a sample data set of the file *server.data*.

www.atv.tuwien.ac.at§Apache/1.3.12 (Unix) PHP/4.0.4pl1 tomcat/1.0§80

Table 6.2: Sample data set of the file *server.data*

The data entries in this file are separated by the separator '§'. Again a perl script parses this file and performs some adjustments on the data. The first entry is the name of the host, the second entry is the Web server and the according version, the underlying operating system in the brackets and the scripts enabled. The last entry is the port, via which we accessed the host. The script now queries the table *dimension* for the domain_ID and stores the Web server including the version in the table *server*, the operating system into the table *OS* the modules installed on the Web servers, into the table *modules* and the port into the table *ports*. Afterwards, it stores the respective IDs of the new entries into the table *domain*. At the dimension tables it always queries the database if there is already such an entry to avoid a primary key constraint violation.

A second script fills the database with the information gathered from the according WHOIS server. It queries the table *domain* and makes a request from the according WHOIS server (which is the RIPE server for the '.at' domain) for each domain with the top level domain '.at'. It stores the registered organization or person into the table *owner* and the zip code into the table *address*.

Another script fills the columns *city* and *country* of the table *address* based on the according zip code. The relation between the zip code and the additional address information is gathered from a zip code catalog of the Austrian Post AG[1].

6.3 Step 3: Filling the Austrian Page Data

Further scripts are used to fill the according pages to the domains stored in Step 1 into the database. The input of these scripts are gathered from the file *stats.data*. The file is parsed to evaluate the additional data for each page on a specific domain. The Web host, the URL, the size of the page, MIME type with the according character set, the last update date of the site, and the number of internal, external, and mail links are stored in this file. The number of internal links arises from all links pointing to the same domain as of the host. External links are all the other links. Mail links are the amount of the appearing mail addresses of the specific page.

The name of the page and the file extension is extracted from the URL. Retrieving the page name out of the URL is a tricky process, because not every URL is as we expected. One might think that this is easy, but we have to mind a lot of different things. First, we have to consider the parameters. That would be easy if the delimiter of the file and its according parameters is always the same namely the question mark. However, in reality there are various combinations of these characters used as delimiters, parameters, files, and file extensions. Second, we have to consider that the path given may also contain question marks, points and other characters that will mess up the parsing algorithms. Due to these problems we studied the URLs and implemented a script checking the sites, which consider a lot of different types of URLs.

The date is parsed as well and adjusted as described at Step 2. Afterwards, it is possible to store the name of the page, the size, the URL, the date of the last update, and the number of links occurring on the page into the table *pages*. After querying the table *time* and the table *filetype* with the according data we

[1] `http://www.post.at/content/online_service/download/online_service_download_plzverz.html`

can also save the additional IDs gathered from these tables into the table *page* to create the join.

6.4 Step 4: Filling the Link Domains

This section describes the first step of the feeding process of the link data, which is all data gathered from Web sites pointed by Web sites in the Austrian Web space. Because of the huge amount of link data, we do not query the WHOIS or DNS servers to get more information about these hosts. Due to their size and growth rate these tables involved are the most critical ones and it is very important to make these scripts as performant as possible. For this reason we developed an algorithm, which fills a temporarily table where the hosts with the most incoming links are stored. During the feeding process of the table *link_domains* the temporarily table *temp_link_domains* is filled by counting the occurrence of the linked hosts. Every 1000 entries the script deletes the entries which are not under the top 30 counted. If a linked host reaches a given number of 'hits', this process is stopped and all linked hosts in this temporarily table with less then a given number of 'hits' are deleted. Hence only the linked hosts with the most hits remain. The remaining process of filling the table *link_domains* is quicker because the script first queries the temporary domain to check if this host has been already entered before. Only when the host is not in this table, the script has to query the table *link_domain* that can take a long time.

The input data is stored in the file *links.data*. It contains the URL of the page containing the link, the host name, where the links points to, the number of links pointing to the host from this page, and the total number of external links on this page. At the moment we have about 453.600 entries in the table *link_domains*. The feeding script was running over weeks and it is very hard to estimate the total time it takes because the script is getting slower and slower. The more data entries in the table, the more data entries have to be read taking a considerable amount of time to check whether a host is already in the table.

6.5 Step 5: Filling the Link Pages

Because of time constraints we did not yet fill the database with the gathered pages where the links point to. We just stored the linked domain and the number of links. For this reason, the process of filling this table is just to build the connection between the table *page_to_page_links* and the according table where the linked host is stored. If it is a link pointing to a host within the Austrian Web space, the connection is established to the table *domains_copy* (domain_ID is stored), otherwise if the link is pointing to a 'foreign' host, the connection is established to the table *link_domains*(link_domain_ID is stored).

The scripts are able to handle data including the link pages. After the next run (crawling the Austrian Web space) we will store this information into the database too. Feeding this table is very time critical, because the growth rate of the table is very high and the more entries are stored, the more time does it take to store a new entry.

6.6 Step 6: Filling the table *page_to_page_links*

As described above we have not yet stored the pages where the links point to, into the database. For future feedings the connection between the table *pages* and the table *link_pages* is established by filling the connecting table *page_to_page_links*. Again, the base data is gathered from the file *links.data*. The script now queries the table *pages* for each page occurred in the file by searching for the right URL. Afterwards, it queries the table *link_domains* for the host linked to. It stores the key of the table *pages* (page_ID) and the key of the table *link_pages* (link_page_ID) together with the protocol of the links into the table *page_to_page_links*. One data entry is stored for each link.

This is the most time critical process of feeding the database. At the moment we have already more than 40 Mio entries in this table. At a status of about 20 Mio entries it takes about one hour for feeding 9000 entries. It is easy to calculate that this feeding process took us several weeks.

6.7 Step 7: Manual and additional adjustments

Several steps have to be done to adjust the data manually. We did not implement a script for these tasks because sometimes it is easier and quicker to do some processes manually. In the following paragraph I will briefly describe in which tables we had to adjust the data.

- *domains*: It emerged during the feeding process that the members of the level sub_sub_domain are far too many. The OLAP tool we used has a limit of 64000 members per level. Due to this fact we had to implement a grouping level. To find the several member entries in the DWH, we decided to group them alphabetically.

- *server*: The server information gathered from each Web server is very often inconsistent. In other words the same server can be spelled differently. For example, the Netscape server is once spelled as 'Netscape Enterprise' and the other time 'Netscape-Enterprise'. The aggregation of the OLAP tool would handle these names as two different Web servers. This task could just be done by orthographical algorithms but in our case it is much easier and time saving to do it manually. A second task, done by hand, was finding the right producer of the Web server to obtain a correct hierarchical grouping. Most of the servers can be easily found in the Web.

- *owner*: The owner information gathered from the WHOIS server is also very often inconsistent. We adjusted the entries, which are apparently spelled differently. We also had to group the member entries alphabetically due to the large number of entries.

- *filetype*: As described in Section 5.2 the MIME types based on the extensions of the files has to be filled manually to obtain the hierarchy type and sub-type of the file.

The feeding process of the database is the most time consuming step of our project. For this reason it is very important to focus on the performance of these script. Some tasks in this process are always improvable. For example, when we

think of the task parsing a link to extract the page or the file extension of the page. We have to consider that there are no rules designing a Web site. Hence we always have to adapt these extracting scripts.

6.8 Summary

In this chapter I described the feeding process of the database, which is a big part of building a Data Warehouse. It is worth to attach importance to this part because it involves rather time critical processes.

The data model of our project can be divided into three parts. The first part arises from all tables containing data referring to the Austrian Web sites. The second part consists of tables representing the information where the Austrian Web sites links to. The table connecting these parts represents the links stored. In Section 6.1, 6.2, and 6.3 I described the feeding process of the data referring to the Austrian Web data. In Section 6.4, and 6.5 I described the process of feeding the linked Web sites. Section 6.6 describes the feeding process of the links. In Section 6.7 I showed the tasks which had to be done additionally and in some cases manually.

Chapter 7

The Data Warehouse

After the data has been stored in the database, a multi-dimensional array structure, called a data cube, is built to aggregate the measures. No matter whether the data is actually stored in a flat relational DBMS using a dimensional design, such as the star or snowflake models, or whether a multi-dimensional DBMS is used. The data cube, defined by a set of dimensions and measures, is the essential part of a Data Warehouse, a technology that provides fast access to data in a Data Warehouse. The data is separated into two categories, namely the *facts* containing the measures and the *dimensions*. Facts is the information that is to be analyzed, with respect to its dimensions. In this section I will briefly describe the main characteristics of DWHs in general. To keep it short I will not be able to address issues of DWH design and different types of data models used for subsequent analytical processing in detail, but refer to the depth of literature on DWH design for these issues, e.g. [37, 38]. Further a Data Warehouse is a good tool to make analysis over time with a time dimension. Data is not updated, but added to a Data Warehouse to get data across several time periods. In our case that would be the different snapshots of the Austrian Web at different times. See [39] for a detailed description of time-related aspects in DWH maintenance. With the aid of OLAP (on-line analytical processing) tools, it is possible to drill-down, roll-up, slice and dice, to view and analyze the Web data from different perspectives, derive ratios and compute measures across many dimensions.

There are a lot of different definitions for OLAP. I chose one, which in my

opinion, best covers the main concepts of this technology. The definition is provided by the project *OLAP Report* [45]. It is a reasonable and understandable definition of the goals, OLAP is meant to achieve.

In this report OLAP is summarized in just five key words namely *Fast Analysis of Shared Multidimensional Information.*

Fast means that the system is targeted to deliver most responses to users within a few seconds. Pre-calculations, which is the process of calculating the aggregations make it possible to achieve this speed with large amounts of data. The aggregations are calculated during the processing of the cube before the user started to work with it. Certainly the aggregations need to be saved and hence need data space. In the OLAP tool you can choose to make your own ratio of saving data space or make more aggregations to get a cube for faster analyses.

Analysis means that the system can cope with any business logic and statistical analysis that is relevant for the application and the user, and keep it easy enough for the target user. The analysis functionality should be provided in an intuitive manner for the target users. This could include specific features like time series analysis, goal seeking, ad hoc multidimensional structural changes, non-procedural modeling, data mining and other application dependent features. These capabilities differ widely between products, depending on their target markets.

Shared means that the system implements all the security requirements for confidentiality and, if multiple write access is needed, concurrent update locking at an appropriate level. The system should be able to handle multiple updates in a timely, secure manner. This is a major area of weakness in many OLAP products, which tend to assume that all OLAP applications will be read-only with simplistic security controls.

Multidimensional This is a base requirement of OLAP systems. The system must provide a multidimensional conceptual view of the data, including full

support for hierarchies and multiple hierarchies.

Information OLAP systems should handle all the relevant data for answering a specific problem. Certainly the capacity of various products differ greatly. The largest OLAP products can hold at least a thousand times as much data as the smallest.

As mentioned before, the multidimensional data cube has numerous dimensions. Each dimension is structured by a hierarchical concept to facilitate generalization and specialization. Building this data cube allows us, using the OLAP tools, to drill-down, roll-up, slice and dice, to view and analyze the Web data from different perspectives, derive ratios and compute measures across many dimensions. The *drill-down* operation can be used, for example, to navigate from the top-level domains to the sub-level domains. The inverse *roll-up* may be used, for example, for the aggregation of total links from hosts, which are located in Graz (a city) compared to all links from hosts located in Styria (the respective county). This is a *roll-up* by summarization over the address hierarchy. The *slice* operation defines a sub-cube by performing a selection on, for instance, *domain = .ac.at* on the dimension *domains*, to get all information concerning the educational Internet domain in Austria. The *dice* operation defines a sub-cube by performing selections on several dimensions. For example, a sub-cube can be derived by dicing the cube on three dimensions using the clause, *county = 'Vienna' and operating system = 'linux' and Web server = 'apache'.*

These OLAP operations assist in interactive and quick retrieval of 2D and 3D crosstables and charttable data from the cube, which allow quick querying and analysis of a very large Web data storage. The quick retrieval is possible due to precalculated summary data called aggregations. Aggregations are created for a cube, by processing it before end users access the cube. The results of a query are retrieved from the aggregations, the cube's source data in the Data Warehouse, a copy of this data on the analysis server, the client cache, or a combination of these sources. An analysis server can support many different cubes, such as a cube for Web hosts, a cube for pages, a cube for links, and so on.

I picked out some data mining functions, which I briefly describe.

- *Class description:* Class description provides a concise summary of a collection of data and distinguishes it from others. The summary of a collection of data is called class characterization; whereas the comparison between two ore more collections of data are called class comparison. For example, the amount of links from a specific Web server of a specific domain can be summarized by a characteristic rule. Another example is to compare the availability of links of two different Web servers.

- *Association:* Association is the discovery of association relationships or correlations among a set of items. For example, as we can see in Section 9.1 there is just one specific type of video files ('Quicktime') stored on a certain type of Web server ('Stronghold Web server').

- *Time-series analysis:* Time-series analysis is to analyze data collected along time sequences to discover time-related interesting patterns, characteristics, trends, similarities and differences. In our case this is the comparison between different runs.

In the following sections I will describe the data cubes and their dimensions used in our project.

Every cube has a schema, which is the set of joined tables in the Data Warehouse from which the cube draws its source data. The central table in the schema is the fact table, the source of the cube's measures. The other tables are dimension tables, the sources of the cube's dimensions.

Each cube dimension can contain a hierarchy of several levels to give the possibility to drill up and down through the data. Hence it is possible to give a good overview of a dimension as well as a detailed insight of selected dimension members.

There are several dimensions used in different cubes, hence I described the dimensions in detail when they are mentioned first. In Section 7.1 I will describe the cube *WebHosts* representing the hosts of the Austrian Web space with additional data represented by several dimensions described. The Section 7.2 contains the information about the cube *Pages*, representing data of the Web pages in the Austrian Web. In Section 7.3, and 7.4 I describe the cube

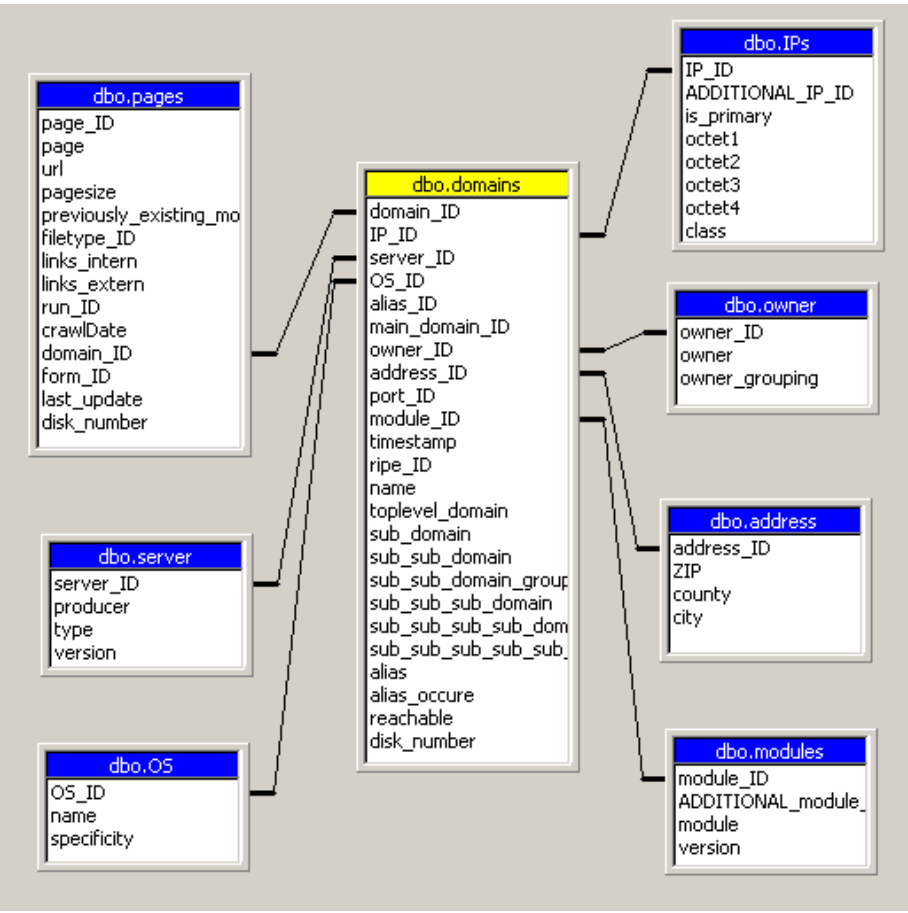


Figure 7.1: Structure of the data cube *WebHosts*

AllLinks, which provides analysis about the links occurring in the Austrian Web space pointing to so called 'foreign' pages, and the cube *AOLALinks*, providing analysis about the links within the Austrian Web space.

7.1 Data Cube *WebHosts*

With this cube it is possible to analyze all the data concerning the hosts. The structure is a classical star schema, which can be seen in Figure 7.1. The table in the middle of the figure is the fact table, which contains the measure column. The other tables are the different dimension tables.

The according dimensions will be described in the following section.

- *Host_Server*:
  This dimension contains the information about the Web servers of the Austrian Web sites. The data is structured hierarchically by producer, product, and version, which can be seen in Figure 7.2. This is not a critical dimension because the number of entries remain within a manageable scope. This dimension is already more or less stable, which means that filling more data into the Data Warehouse does not mean that this dimension will grow accordingly. A list of the producers, Web servers, and their versions are shown in Table A.1 in the appendix of this thesis.

- *Host_OS (Operating System)*:
  The dimension *OS* contains the information about the underlying operating system of the Web sites. Due to the fact that this information comes from the Web server of each host, we can not guarantee the accuracy of this information. Sometimes the servers provide wrong information about the underlying operating system intentionally to mask themselves. The two-level structure of this dimension is described by the name and the specificity of the operating system. The growth rate of this dimension according to the feeding process is similar to the dimension *Server* described above. It is even a bit lower because we do not have the different versions of the operating systems.

- *Host_IPs*:
  With this dimension it is possible to differentiate between different networks, which are characterized by their according IP addresses. For example, a big company often has their own network domain, which is characterized by the first octet of the IP address. This dimension nearly grows according to the number of entries in the table *domains*. In other words there is one IP entry for each entry in the table *domains* minus the aliases (they have the same IP address) and the hosts not reachable, plus the additional IPs of the hosts. At the moment there are 86 different members of the first level, which specifies the first octet of the IP address.

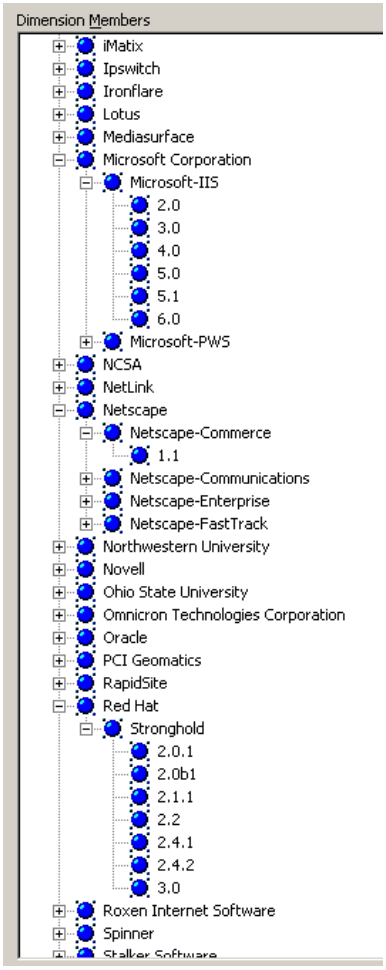


Figure 7.2: Structure of the dimension *Host_Server*

- *Host_IPClass*:
  This dimension is also based on the table *IPs*. But this dimension will not grow when more data is filled into the DWH because there are just three different classes, which classify the IP addresses into different sizes of networks. For example, it gives us the possibility to slice the cube into a smaller cube containing just Web sites appearing in a big (class A) network, eliminating most of the private Web sites.

- *Host_Owner*:
  The organizations or persons stored in this dimension are registered at the respective WHOIS server. As we did at the dimension *domains* described in Section 6.7, we also had to group this dimension alphabetically due to the large amount of members. The growth rate of this dimension is more or less the same as of the dimension *domains*. There are a bit fewer entries because there are companies or persons, which register more than one host.

- *Host_Address*:
  This dimension gives us the possibility to drill up and down from the level of city to the county. It is possible to compare the different counties as well as making aggregations of different cities to name just a few of the large amount of possibilities. This dimension is also filled with data from the WHOIS server. So we have to keep in mind that we do not deal with the physical destination of the Web servers, but rather with the address entered at the WHOIS server for each host.

- *Host_Aliases*:
  The underlying table of this dimension is the fact table of the cube. As described in Section 6.1 the aliases of a Web host are stored in the table *domains*. This dimension gives us the possibility to slice the cube into a part of it just containing aliases. Hence we can discover some characteristics of the Web hosts concerning aliases.

- *Host_Modules*:
  This dimension allows analysis of the different kind of modules installed at the Web servers. There are two levels to drill down this dimension.

First level contains the name of the module and the second one shows the version of the modules.

- *Host_Filetype*:
  This is one of the most interesting dimensions and will be used at various cubes. The underlying table *filetype* is described in the Section 5.2. The structure of this dimension is a so called snowflake structure because the data is based on a table not directly connected to the fact table. This dimension gives us the possibility to incorporate the types of the files stored on each host. For example, we are able to group the Web hosts by their most types. This dimension is based on the MIME types gathered from the Web servers. Due to this fact we can not guarantee the accuracy of this information. As described in Section 5.2 there are a lot of inconsistencies between the MIME types gathered from the Web servers and the file extensions of the the according files. For this reason we implemented another dimension based on the file extensions described beneath. The growth rate of this dimension is rather high. But due to the fact that we store each different file type just once, the rise of the growth rate is getting lower, the more file types we have stored. We just have to consider that the amount of different MIME types will always grow or at least change because the information technology is growing and changing, implementing new software, which uses new MIME types.

- *Host_FileTypeExtension*:
  This dimension is more or less the same as the *Host_Filetype* dimension with the big difference that it is based on the information of the file extensions as described in Section 6.7. It is quite a big challenge to extract always the right file extension from the URL. The MIME types based on the file extensions are very often completely different to the MIME type of the *Host_Filetype* dimension. Not all the file extensions could be assigned to MIME types. Therefore we partitioned the unassigned records into groups representing the number of occurrences like `1`, `2-10`, `11-100`, `101-500`, `501-1000`, and `1000+`. This grouping is shown in Figure 7.3.

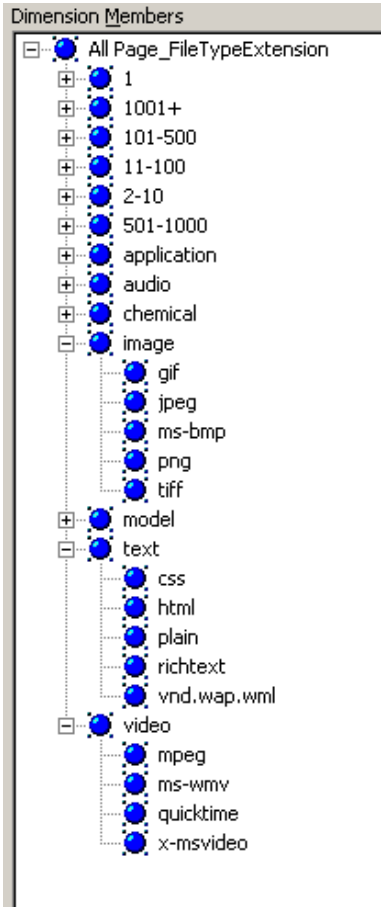


Figure 7.3: Structure of the dimension *Host_FileTypeExtension*

- *Host_LastChangeClientDate*:
  This dimension gives us the possibility to categorize the Web pages on the basis of their date of the last modification. The underlying data are the dates of the last update for each page and is gathered from the Web servers. As mentioned before, the information from the Web servers are not necessarily accurate. This dimension gives us various kinds of possibilities for analysis. For example, it is possible to compare the file types of the pages with the dates of their last update, or to extract the Web sites with the most recent updates of their pages.

7.2 Data Cube *Pages*

This cube is implemented for analyzing the data concerning the Web pages of the Austrian Web. Measures calculated are the sum of the page_ID, which is in fact the number of pages and the accumulated size of pages selected. Hence the fact table is the table *pages* in the middle of the surrounding dimensions described in the following paragraph. The structure of this cube can be seen in Figure 7.4. Of course these schemes showed in this thesis are just sample schemes showing the most important possibilities. In fact the cubes are processed only with the dimensions needed for a given problem. It would be much more time consuming to process the cube containing all dimensions possible. In Section 8.1 I will describe the OLAP tool, which is our basic tool for building the Data Warehouse.

- *Page_Domains*:
  This dimension contains the different Web hosts stored in the table *domains*. As mentioned before, the dimension is structured into the several levels of the domains. Figure 7.5 gives us a good example showing members at each level of this dimension. We have to consider that there are a lot more sub level domains in the Austrian Web than the maximum allowed number of 64000 members for one level, a restriction imposed by the MS OLAP tool. Due to this fact, we had to add a grouping level, which is structured alphabetically, which however, allows a more efficient

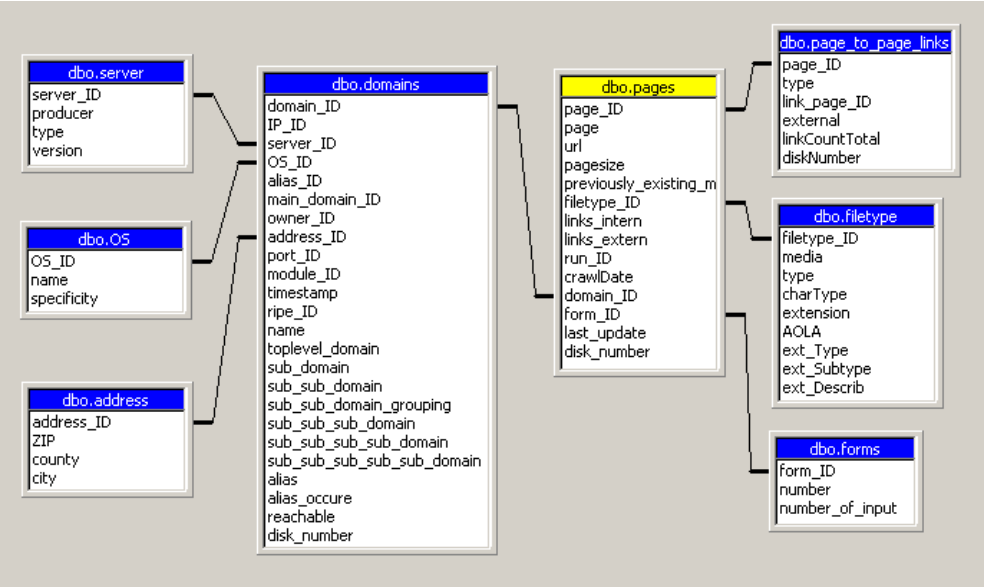


Figure 7.4: Structure of the data cube *Pages*

interactive analysis, as a drill-down resulting in an extension into more than 64000 branches would not be useful.

It gives us the possibility to qualify the measures (e.g. number of pages) by choosing a certain domain, as for example, aggregating the number of pages on the domain 'tuwien.ac.at'. Certainly it will make more sense using more dimension at the same time, for example, to count the pages of the MIME type 'image/jpg' on this domain, etc.

- *Page_Filetype*:
  This dimension is very similar to the dimension *Host_Filetype* described above. The underlying table is as above the table *filetype*. The only difference is that it is used in another cube and hence has a different structure. This dimension has in contrast to the dimension *Host_Filetype* a star schema because the table is directly joined to the fact table. This can be seen in Figure 7.4.

- *Page_FiletypeExtension*:
  This dimension has also been already described in detail with the cube

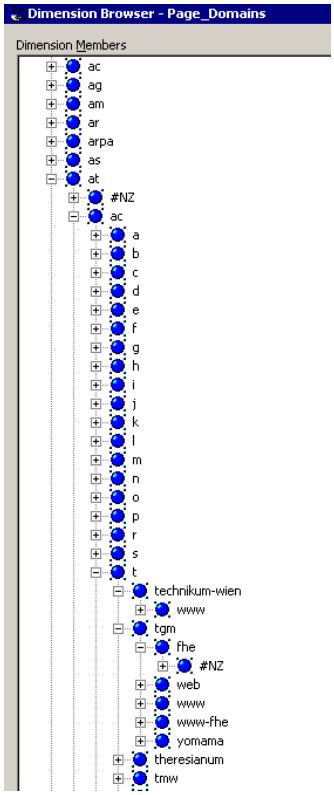


Figure 7.5: Structure of the dimension *Page_Domains*

WebHost. Please refer to Figure 7.3 to see the structure of this dimension. It gives us the possibility to analyze the various MIME types based on the extensions of the files combined with a lot of possible constrains of other dimensions.

7.3 Data Cube *AllLinks*

This cube is implemented to perform the analyses concerning the links data and link structure. The structure of the cube can be seen in Figure 7.6. The table in the middle of the figure is the fact table, which represents the links pointing from the Austrian Web sites, stored in the tables *domains* and *pages*, to the 'foreign' Web sites stored in the tables *link_pages* and *link_domains*.

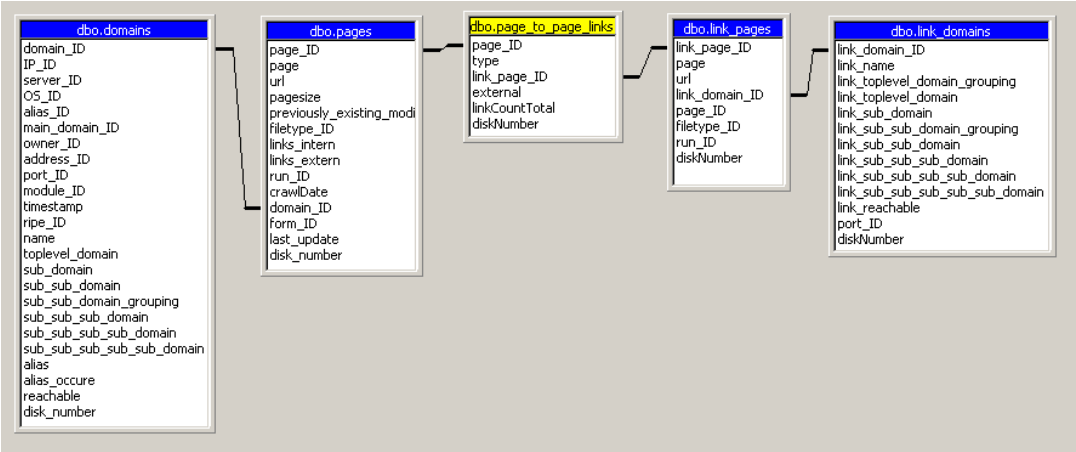


Figure 7.6: Structure of the data cube *Links*

The according dimensions will be described in the following section.

- *AllLinks_Domains*:
  This dimension contains the information about the Web sites of the Austrian Web. As mentioned before, the dimension is structured into the several levels of the domains, which can be seen in Figure 7.5. For a detailed description, please refer to the Section 7.2.

- *AllLinks*:
  The dimension *AllLinks* contains the information about the so called 'foreign' Web sites. These Web sites are linked by Austrian Web sites and are themselves not in the Austrian Web space. The dimension is structured as the dimension *AllLinks_Domains* except that we had to modify the grouping level. Due to the fact that these tables contain much more data then the according tables containing the Austrian Web data, we grouped the third level domains by the first two characters of the string, which can be seen in Figure 7.7. Therefore, we can break down the next level to prevent a member overflow.

Of course a lot of the dimension described in the sections before, can be added into this cube to be able to perform several combined analyses.

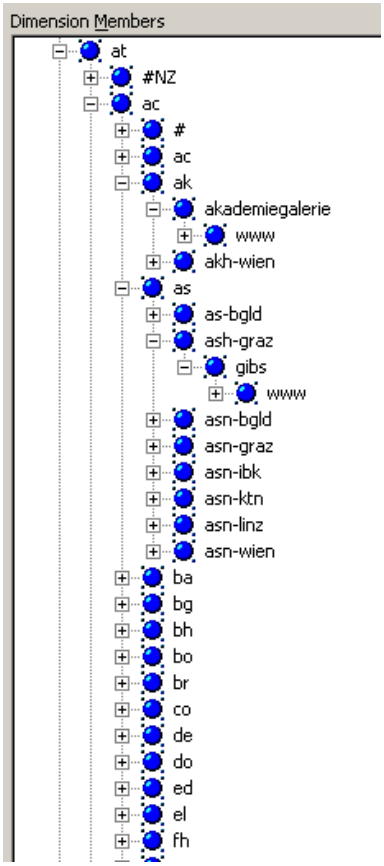


Figure 7.7: Structure of the dimension *AllLinks*

7.4 Data Cube *AOLALinks*

This cube is very similar to the cube *AllLinks*. It is also implemented to perform the analyses concerning the links data and link structure. Whereas this cube represents the data of the Austrian Web sites connected to other Austrian Web sites. As you can see in Figure 7.8 the Austrian Web sites are again stored in the tables *domains* and *pages* whereas the links are pointing to the table *link_pages*, which is in turn connected to the tables *domains_copy* and *pages_copy* that are copies of the former tables. This cube structure was implemented to avoid cyclic links.

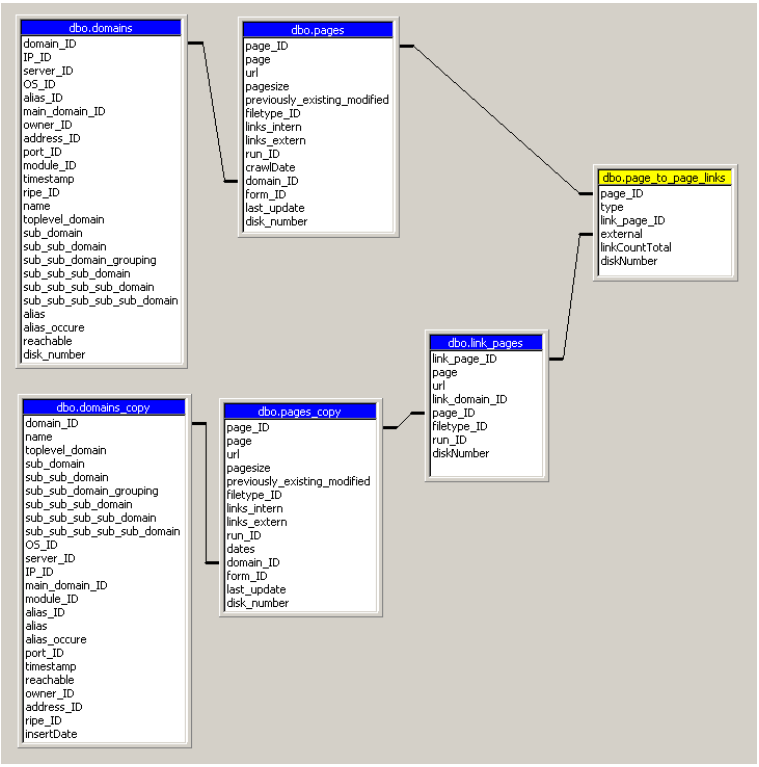


Figure 7.8: Structure of the data cube *AOLALinks*

The according dimensions will be described in the following section:

- *AOLALinks_Domains*:
  This dimension contains the information about the Web sites of the Austrian Web. For a detailed description, please refer to the Section 7.2.

- *AOLALinks*:
  The dimension *AOLALinks* contains the information about the linked Austrian Web sites. These Web sites are linked by other Austrian Web sites. The dimension is structured as the dimension *AOLALinks_Domains*.

Again, a lot of the dimensions described in the sections before, can be added into this cube to be able to perform several combined analyses.

7.5 Summary

In this chapter I described the cubes and dimensions we used in our project. In Section 7.1 I described the cube *WebHosts* representing the hosts of the Austrian Web space. In Section 7.2 we talked about the cube *Pages*, representing data of the Web pages in the Austrian Web. The cubes *AllLinks*, and *AOLALinks* provide analysis about the Web sites pointing to the 'foreign' pages and to the Web sites linked within the Austrian Web space. Of course there are a lot more combinations of cubes and dimensions possible to provide further analysis about special cases. One of the biggest advantages of our project is the possibility to modify and change the parameters in order to provide a solution for several different tasks.

Chapter 8

Interface Design

In the first section of this chapter I will primarily describe the OLAP tool we used. I will give an overview of the possible analyzing functions combined with various screenshots to make it clear. In Section 8.2 I will briefly describe the analysis functionalities in MS Excel.

8.1 Analysis Manager from Microsoft

The underlying database of our project is a Microsoft SQL Server 2000 database, which is sufficient for the requirements of our project. For simplicity we decided to use the Analysis Services as our OLAP tool, which is included with the SQL Server 2000. The easiest way to generate graphics for our analysis is to use the data import function of Microsoft Excel. In the following section I will briefly describe the steps necessary to get a graphic table showing the contribution of operating systems over several domains, starting from the point of a fully loaded database.

- *Creating a new database on the analysis server and setup the connection to the source database*
  After creating a new database in the Analysis manager, you have to setup a new data source connection. The data source is located in the SQL Server. We created an ODBC connection before and choose this connection as our data source now.

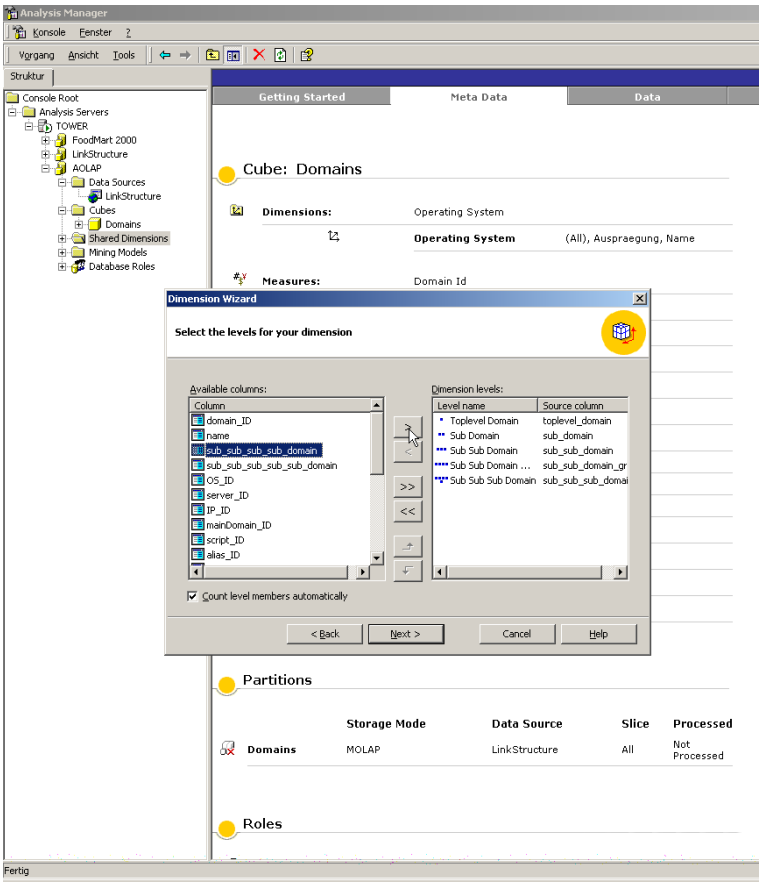


Figure 8.1: Wizard to create the dimension

- *Creating the dimensions*

  After we open the Dimension Wizard, we define the hierarchy of the dimension in the first step. We choose a Star schema because this dimension is based on just one database table namely *domains* selected at the next step. After defining the levels of this dimension in the next step, which can be seen in Figure 8.1 you can choose some advanced options. They are not so important and can be omitted for our intentions. At the last step we label the dimension namely 'Domains'.

  We create the second dimension *OS* for the operating system similarly.

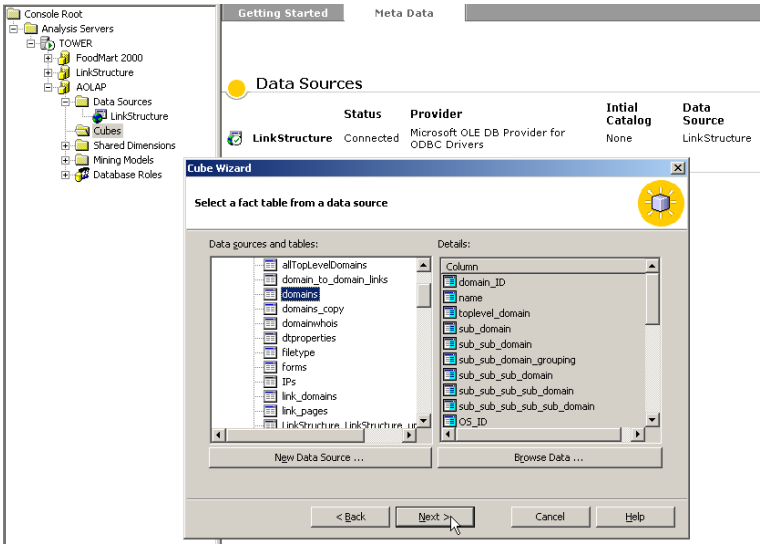


Figure 8.2: Wizard to create the cube

- *Building the cube*
  When opening the Cube Wizard, you have to choose the fact and the dimension tables, which can be seen in Figure 8.2.

  At the next step you have to select the measure columns of the fact table. In our case we selected the column *domain_ID*. Certainly it is possible to select more than one column. Further we choose our previously defined dimensions. At the last step of the Cube Wizard you have to label the cube.

- *Cube Editor*
  In the Cube Editor shown in Figure 8.3 you can visualize the structure of the cube and if necessary you can change the joins of the tables and insert new tables.

  Maybe the Cube Editor is the most important part of the Analysis Manager because here you can adapt, create and change the cubes and dimensions beneath other functionalities offered. As you can see in Figure 8.4 it is possible to browse the data too. In this window you can handle the slices, dices, roll-ups, drill-downs, drill-throughs, and other data mining

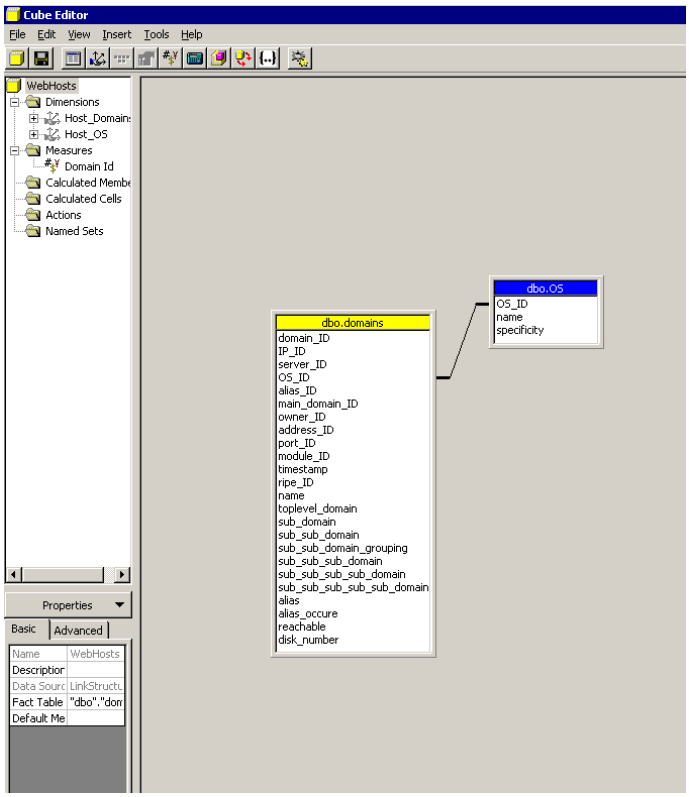


Figure 8.3: Cube editor

techniques supplied by the Analysis Manager.

- *Processing the Data Cube*
  To calculate the necessary aggregations, we have to process the cube before we can browse the data. We open the Storage Design Wizard and select the type of data storage. There are three options:

  – *MOLAP* stores the data of the cube in a multidimensional structure. The aggregations for this storage type will also be stored with the multidimensional data.

    Multidimensional OLAP (MOLAP) storage provides the potential for the most rapid query response times, depending only on the percentage and design of the cube's aggregations. In general, MOLAP is

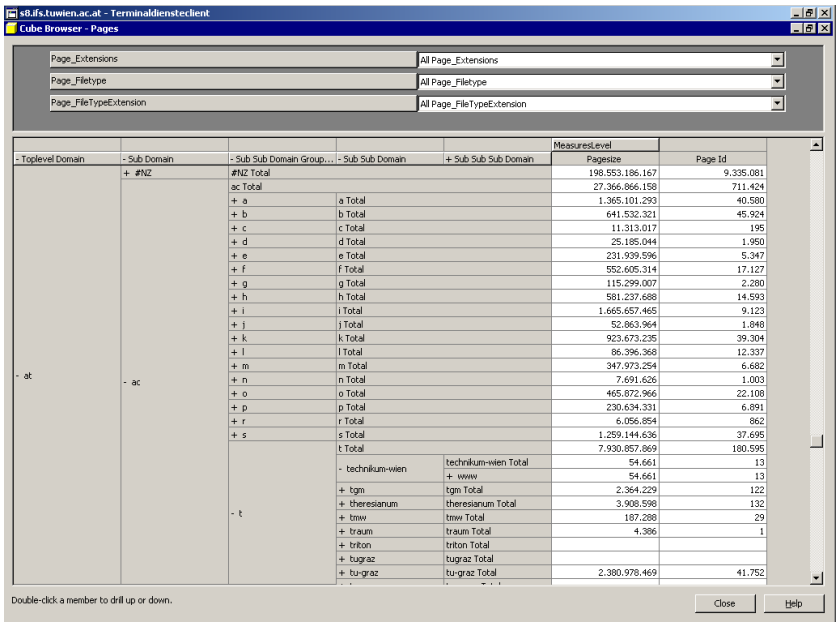


Figure 8.4: Browsing the OLAP cube

more appropriate for cubes with frequent use and the necessity for rapid query response.

– *ROLAP* keeps the data for the cube in the existing relational data store. Aggregations designed for relational OLAP (ROLAP) will also be stored in the relational database, rather than in a multidimensional structure.

  ROLAP query response is generally slower than that available with MOLAP or HOLAP. A typical use of ROLAP is for large datasets that are infrequently queried, such as less recent historical data.

– *HOLAP* stores the data for the cube in the existing relational data store and keeps the aggregations in a multidimensional structure.

  For queries that access summary data, hybrid OLAP (HOLAP) is equivalent to MOLAP. Queries that access base data, such as a drill-down to a single fact, must retrieve data from the relational database and will not be as fast as if the base data were stored in the MOLAP structure. Cubes stored as HOLAP are smaller than equivalent

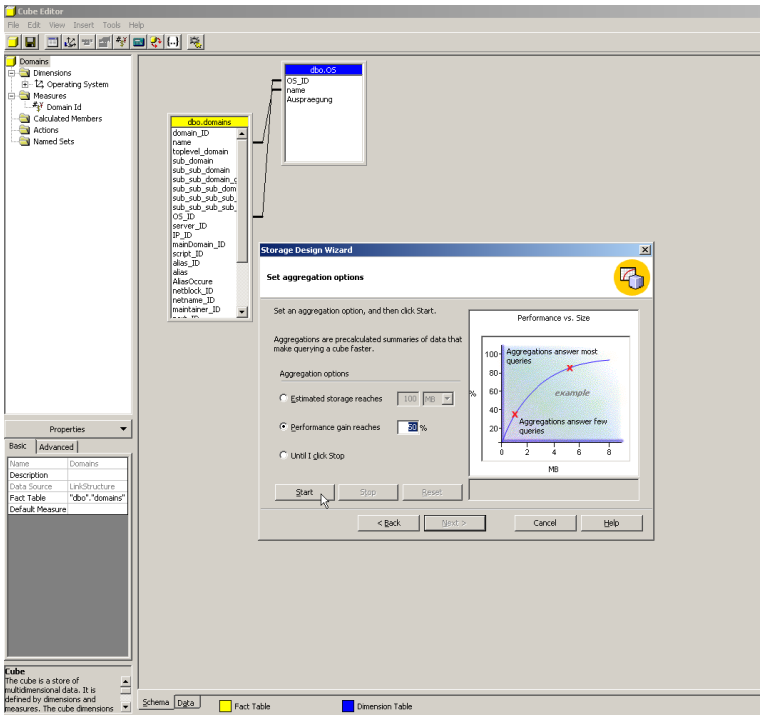


Figure 8.5: Setting of the aggregation options with MOLAP data storage

> MOLAP cubes and respond faster than ROLAP cubes for queries involving summary data. HOLAP storage is generally suitable for cubes that require rapid query response for summaries based on a large amount of base data.

We selected the MOLAP option because it allows for processing the fastest queries. As far as our experience goes, this type of data storage is in the most of our cases the best one. Just when analyzing, using the cube containing the link data, some problems with the data storage occurred. When we drilled down a dimension, showing a large number of members of a level, we got the error message `'There is not enough memory available to display the requested cell set'`. When we have used the ROLAP storing option this problem occurred even more often. Setting the aggregation options, which can be seen in Figure 8.5, is important for the performance of the analysis with the cube. In our experience the best

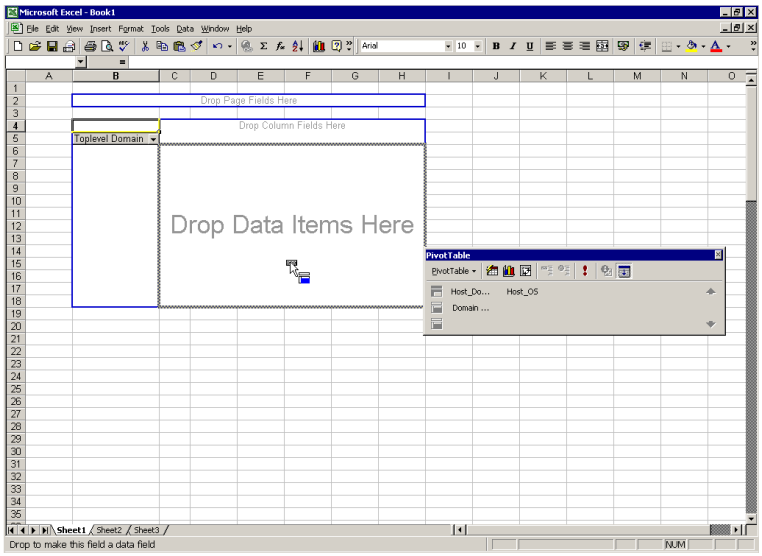


Figure 8.6: Importing the data to Microsoft Excel

ratio of performance versus size of aggregation is at about 60 % to 80 % performance increase depending on the cube and dimensions used. After processing the cube it is ready to browse the data.

8.2 Using MS Excel

Data between the cubes in the Analysis Manager and Excel are channeled with the use of OLE DB for Olap Services. To import the data we choose the menu entry 'Create a new Query' in Excel. Then, we have to select our data source. In the popup there is a tab called 'OLAP-cubes', where we select our cube, created before. This will open a Wizard to create a Pivot table. After selecting the area where we want to place the table, we can add the dimensions into the row and column areas of the table. We add the operating system into the column area, the domains into the row areas, and the measures into the data area where they are calculated automatically. This can be seen in Figure 8.6.

The pivot table in Excel is also a very flexible and easy-to-use functionality. By double click it is possible to drill down or drill up the dimensions used. It

is very easy to visualize the data calculated in a diagram by using the 'Chart Wizard'.

8.3 Conclusion

We used the Microsoft Analysis Manager, described in Section 8.1, because it is easy to combine it with the SQL Server 2000 as they are both Microsoft products. Further the combination with MS Excel, described in Section 8.2, is very comfortable and easy to handle. Due to this fact the cube implemented can remain on the server and we can import the data very easily over a defined connection to the server even we are on a different workstation.

Chapter 9

Analysis of the experimental results

In this chapter I present examples of the analytical capabilities of the AOLAP system. We should emphasize, however, that the current results are based on incomplete crawls of the Austrian Web space, representing data from the first pilot crawl in spring 2001 and a second crawl started in spring 2002. Thus, the numbers provided below can only depict a trend, rather than be taken as confirmed results yet. However, the large amount of data already available at least allows us to analyze the current situation of the Austrian Web space, as well as to obtain ideas of its usage, challenges with respect to its preservation, as well as to discover the benefits of interactive analysis provided by a DWH-based approach. In order to exploit the most important characteristic of such a Web archive, i.e. to analyze its historic perspective and use this as a basis for impact evaluation and trend analysis, a series of snapshots over several years will need to be accumulated in order to facilitate evaluation along the time dimension. In Section 9.1 I will describe a sample analyzing process step by step, by providing the distribution of file-types over the different Web servers. Section 9.2 provides the results of the analysis of the distribution of Web servers over the counties in Austria, whereas in Section 9.3 we can see the distribution of Web servers across the domains. In Section 9.4 we can find some sample analyzing processes about the link structure of the Web.

9.1 Distribution of file-types over different Web servers

In this section I will describe the process of analyzing the distribution of file-types over different Web servers in the Austrian Web space step by step.

We want to ascertain some relations between the file-types of the Web pages and the Web servers used to provide these pages. Let us start by taking a look at the various file formats present in the archive. The number of file types encountered in the Web archive is of high relevance with respect to the preservation of the archive, in order to keep the pages viewable in the near and far future. It also represents a good mirror of the diversity of the Web with respect to the technologies employed for conveying information. All over we encountered more than 200.000 different types of files based on their extensions, and more than 200 different types of information representation when we use the MIME type as the indicative criterium. However, we should stress, that the quality of the information provided this way is very low, as a large number of both file extensions as well as MIME types are actually invalid, such as files with extensions *.htmo*, *.chtml* or *.median, .documentation.* A listing of some of the most important types of files found in the archive is provided in Table 9.1. For a comprehensive overview of almost 7.000 different file extensions and their associated applications, see [46]. While the major part of file extensions encountered are definitely erroneous, they point towards serious problems with respect to preserving that kind of information, as well as the need to define solutions for cleaning this dimension to obtain correct content type descriptors.

Several interesting aspects can be discovered when analyzing the distribution of file types across the different types of Web servers. Therefore we built a special cube with five different dimensions. We used the *Analysis Manager* from Microsoft to built up the Data Warehouse and to make our analysis.

The first dimension called *Page_Domains* contains the domains or Web sites of the Austrian Web space. It is structured hierarchically, reflecting the structure of the domains (top-level, sub-level, etc.) into the levels of our dimension. Therefore we are able to drill through the different levels of each domain. To

MIME type	# Occ.	MIME type	# Occ.
Application/ms-excel	1227	Image/gif	35144
Application/ms-powerpoint	841	Image/jpeg	145200
Application/msword	14799	Image/png	349
Application/octet-stream	9916	Image/tiff	1025
Application/pdf	67976	Image/x-bitmap	426
Application/postscript	5274	Image/other	123
Application/x-dvi	634	Text/css	713
Application/x-msdos-program	1231	Text/html	7401473
Application/x-tar	2189	Text/plain	32549
Application/x-zip-compressed	15314	Text/rtf	2783
Application/other	6985	Text/vnd.wap.wml	2961
Audio/basic	246	Text/other	753
Audio/x-mpegurl	3947	Video/mpeg	983
Audio/x-midi	1777	Video/msvideo	596
Audio/x-mpeg3	3240	Video/quicktime	768
Audio/x-pn-realaudio	5006	Video/x-ms-asf	646
Audio/x-wav	1430	Video/unknown	4
Audio/other	671	Video/other	20

Table 9.1: Selection of MIME types encountered

keep track of the huge amount of different sub-level domains, we included a grouping level ordered alphabetically between the sub-level and sub-sub-level domain.

The second dimension called *Page_Server* contains the information about the Web servers used. The first level of this dimension represents the producer of the Web server like e.g. `'Microsoft Corporation'` or `'Apache Group'`. The second level contains the type of the server like e.g. `'IIS'` or `'Apache'`. The third level describes the version of the servers.

The third dimension called *Page_Filetype* is based on the MIME types gathered from the Web servers, as described in Section 7.2. The dimension is structured hierarchically by *media* (e.g., application, video, or image), followed by the *MIME type*.

As described in Section 7.1, we also defined a dimension namely *Page_FiletypeExtension*, which is more or less the same as the *Page_Filetype* dimension with the big difference that it is based on the information of the file extensions as described in Section 6.7. We associated the file-types to the different MIME types manually. Not all the file extensions could be assigned to MIME types. Therefore we partitioned the unassigned records into groups representing the number of occurrences like `1`, `2-10`, `11-100`, `101-500`, `501-1000`, and `1000+`. This grouping is shown in Figure 9.1.

This separation is necessary due to the fact that the information provided both by the MIME type as well as by the file extensions is prone to errors, and quite frequently these two dimensions do not correspond to each other. Retaining both types of information domains thus provides greater flexibility in the analysis.

Further dimension, added to this cube is *Page_Extensions*, which provide the extensions of the Web pages grouped alphabetically.

After setting up the dimensions and the cube, we can now process the cube. This and other necessary processes executed in *Analysis Manager*, are described detailed in Section 8.1.

9.1.1 Step 1: Setting the Dimensions

The first step of the analyzing process is to set the dimensions necessary for our needs. We drag the dimensions *Page_FiletypeExtension* from the relational schema at the top and drop it on the dimension folder of the cube. As shown in Figure 9.1 we are now able to see all members of the first level of this dimension.

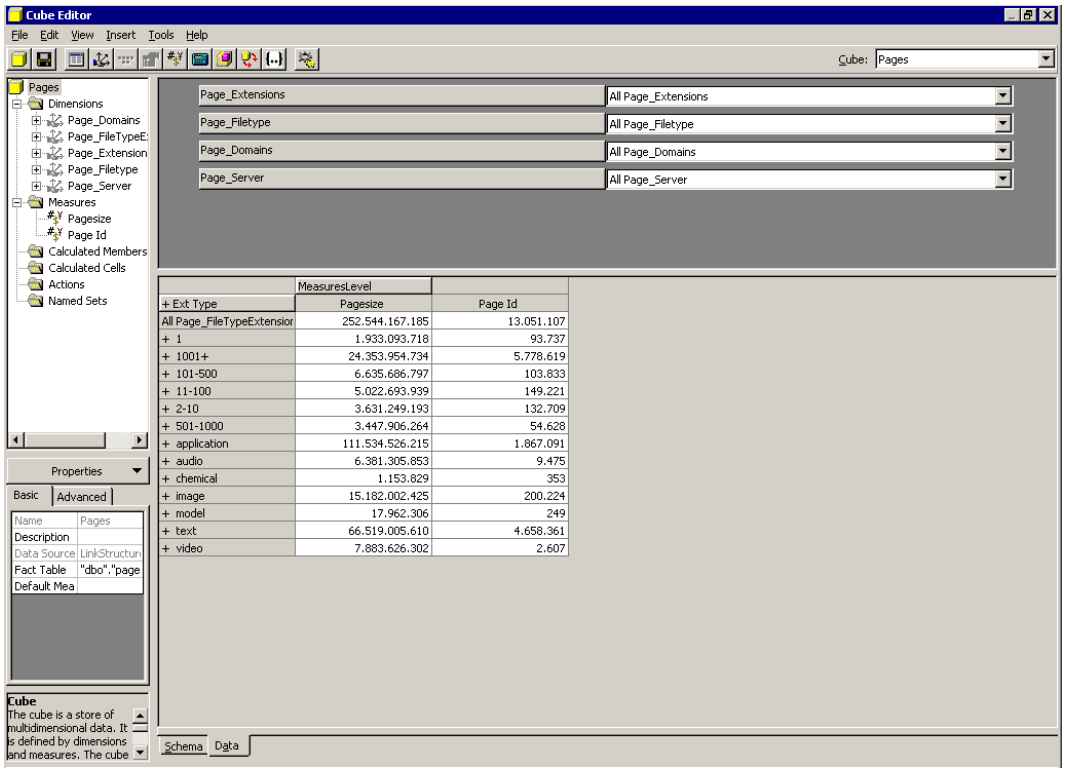


Figure 9.1: Cube showing the dimension *FileTypeExtension*

The data grid in the middle of the cube contains the measure values, in our case these are the accumulated numbers of the page sizes, and the number of pages.

9.1.2 Step 2: Drill down the dimension

As we can see in Figure 9.2 we added the dimension *Page_Server* to the cube and drilled down the dimension members `'Image'` and `'Text'`. At the next level,

we can see that there are several different MIME types provided. We also drill down the *Page_Server* dimension to go into more details at interesting places.

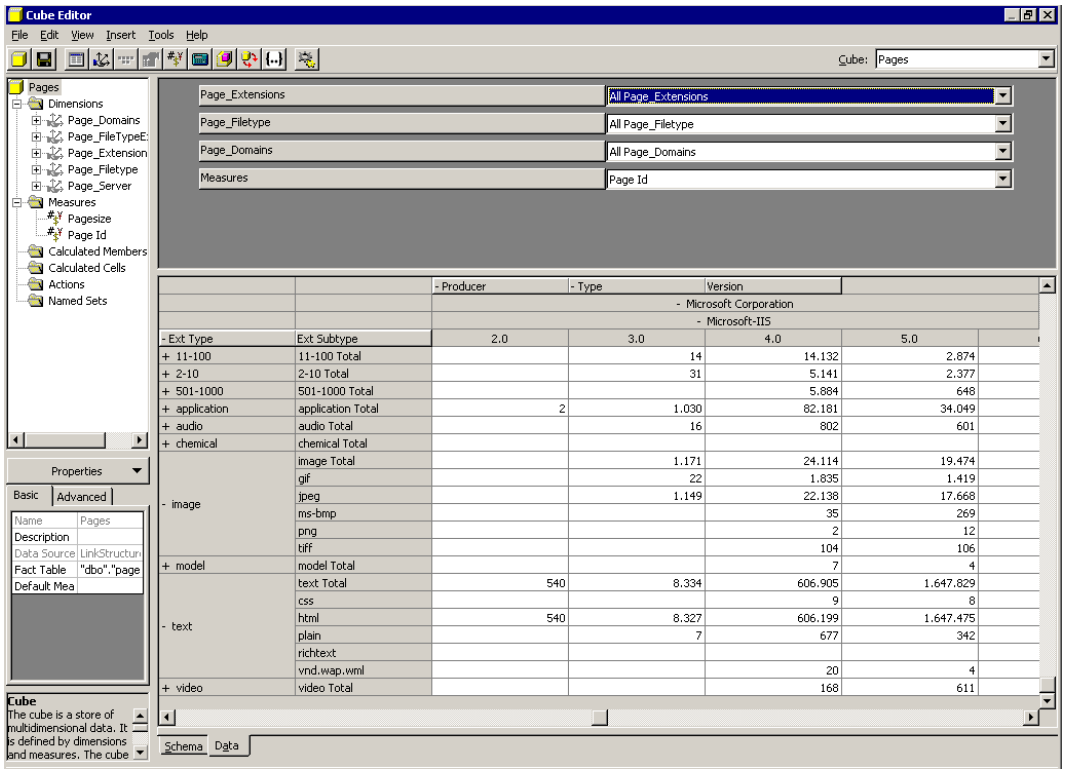
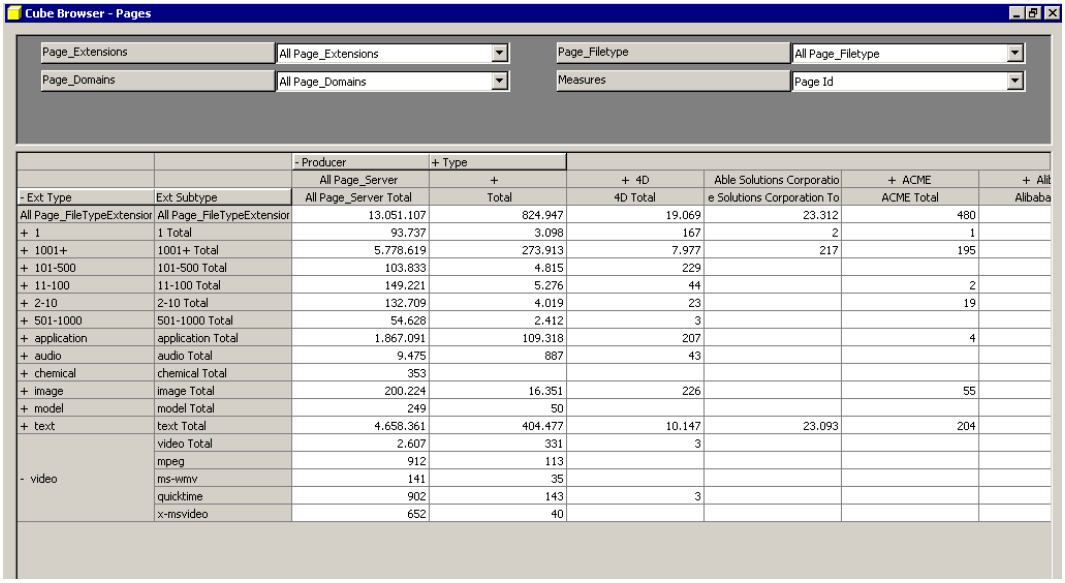


Figure 9.2: A Drilldown of the dimension *Server*

9.1.3 Step 3: Another Drill-Down

Since we are interested in the most prominent way of making available video demos, we decide to take a closer look at these types of files, and servers, which are being used for providing them. As we can see in Figure 9.3 we confronted the video formats with the dimension *Page_Server* and drilled down the video file types.

When we have a closer look at the the dimension *Page_Server*, we find significant differences the way video information is provided with respect to the type of Web server employed. *Mpeg* is by far the dominant format on Apache



		- Producer	+ Type				
		All Page_Server	+	+ 4D	Able Solutions Corporatio	+ ACME	+ Ali
- Ext Type	Ext Subtype	All Page_Server Total	Total	4D Total	e Solutions Corporation To	ACME Total	Alibaba
All Page_FileTypeExtensior	All Page_FileTypeExtensior	13.051.107	824.947	19.069	23.312	480	
+ 1	1 Total	93.737	3.098	167	2	1	
+ 1001+	1001+ Total	5.778.619	273.913	7.977	217	195	
+ 101-500	101-500 Total	103.833	4.815	229			
+ 11-100	11-100 Total	149.221	5.276	44		2	
+ 2-10	2-10 Total	132.709	4.019	23		19	
+ 501-1000	501-1000 Total	54.628	2.412	3			
+ application	application Total	1.867.091	109.318	207		4	
+ audio	audio Total	9.475	887	43			
+ chemical	chemical Total	353					
+ image	image Total	200.224	16.351	226		55	
+ model	model Total	249	50				
+ text	text Total	4.658.361	404.477	10.147	23.093	204	
	video Total	2.607	331	3			
	mpeg	912	113				
- video	ms-wmv	141	35				
	quicktime	902	143	3			
	x-msvideo	652	40				

Figure 9.3: Dimensions of the file type and the Web servers

Web servers, followed by *Quick-time*, which is less than half as popular, but still ahead of various other video formats identified by their MIME type as flavors of *ms-video*.

This is sharply contrasted by the situation encountered at Web sites running the MS IIS Web server.

As described in Section 7.1 we also defined a dimension namely *Page_Filetype*, which is more or less the same as the *Page_FileTypeExtension* dimension with the big difference that it is not based on the information of the file extensions, but on the data provided by the Web servers.

This separation is necessary due to the fact that the information provided both by the MIME type as well as by the file extensions is prone to errors, and quite frequently these two dimensions do not correspond to each other. Retaining both types of information domains thus provides greater flexibility in the analysis. When drilling down the dimension *Page_Filetype*, we can see that the family of *ms-video* and *ms-asf* formats far dominate the type of video files provided on the MS IIS Web server. This format is used by MS Active Streaming (Media) files containing audio and/or video data, compressed with

3rd party codecs. When we take a look at the Netscape Web server we again find a slight dominance of *ms-video* file formats.

When we scrolled the Web servers, we noticed a strange anomaly. The Stronghold Web server, which is the Red-Hat Secure Web server for Linux operating systems, when it comes to video files, provides only *Quicktime* movies which is marked in Figure 9.4. Untypical distributions like this may quite frequently be attributed to artifacts such as a single Web server running a specific system and providing a large amount of files as part of a collection. In order to find out whether this is a generic trend or just an artefact we drill down the dimension *Domains* setting the Stronghold server at the dimension *Server* and the video file types at the dimension *Page_Filetype*. In Figure 9.5 we can see that there are different Web sites using the Server Stronghold, containing regarding the video files exclusively *Quick-time* movies.

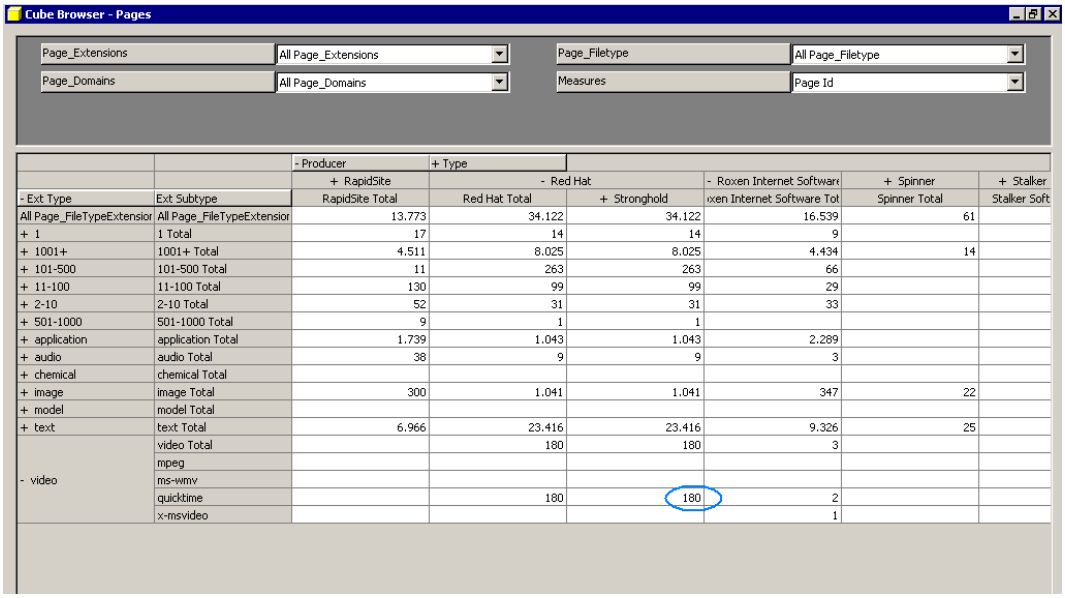


Figure 9.4: A drilldown of the dimension *Page_FileTypeExtension*

Actually, the distribution can be attributed to a sub-group of 10 domains out of several hundred sites using the Stronghold server. Of these 10 sites, however, 9 are closely related to each other and are part of one larger organization providing identical information, thus actually being a kind of mirror of one site. Due to

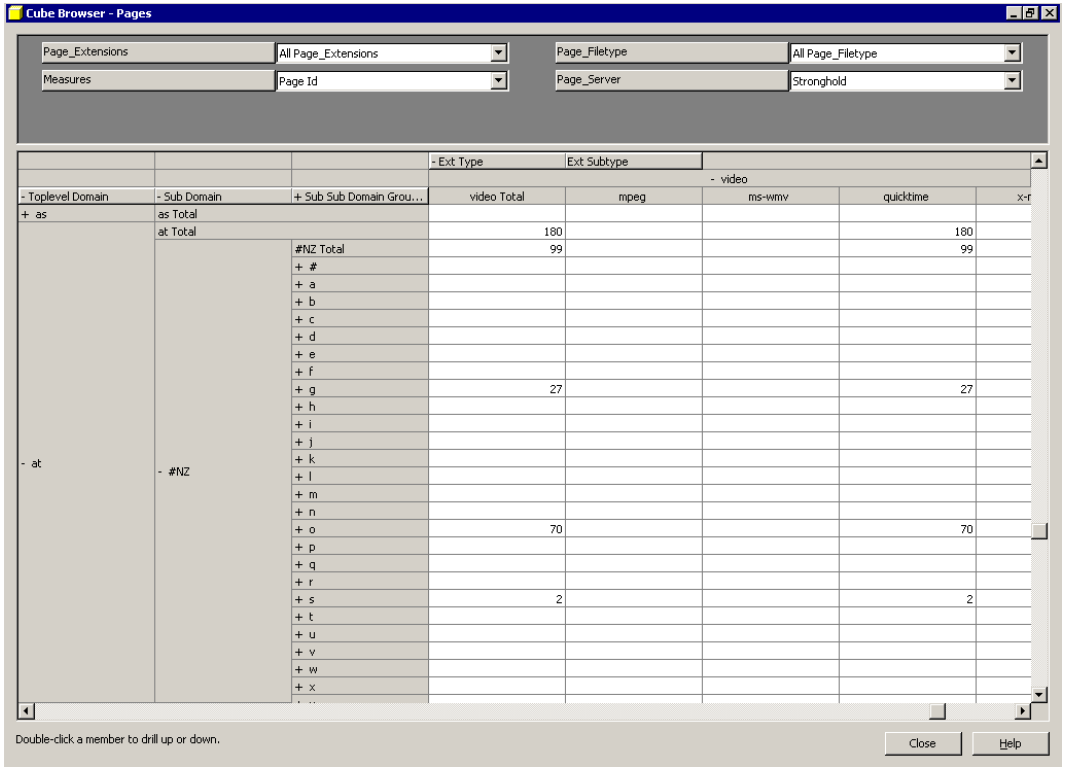


Figure 9.5: Domains storing Quick-time files on Stronghold Web servers

the flexibility of the interactive analysis facilitated by the DWH, these artifacts can easily be identified.

When we drill down the dimension *Page_FileType*, we find a video format identified as MIME type *video/unknown* on Apache servers. By viewing the associated file extension dimension these files were identified to be *.swi* and *.raw* files, the former, for example, being a *swish* data file used in connection with *Flash* animations).

9.1.4 Step 4: Creating diagrams

Now, we want to build a diagram showing the distribution of the file-types over Web servers. First, we have to import the data in Microsoft Excel (this process is described in Section 8.1). Then, we built the pivot table by dragging the data

items (in our case the *page_ID*) into the main part of the table. Afterwards, we drag the dimension *Page_Server* into the left column field, and the dimension *Page_Filetype* into to top column field. Afterwards, we deselect all members of the dimension *Page_Filetype* except the video types. We also constrict the dimension *Page_Servers* so that we just see the entries of the top ten top-level domains containing the Web pages, which can be seen in Figure 9.6.

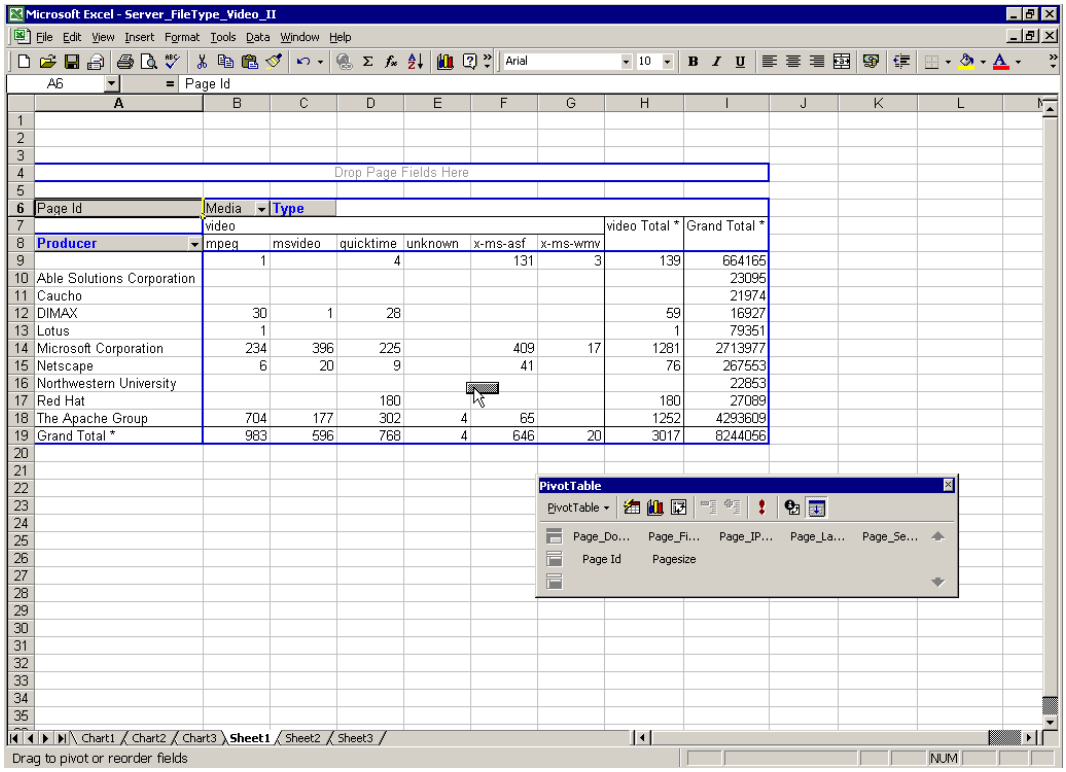


Figure 9.6: The pivot table in Excel

When the pivot table shows the right data entries we want, we can make a diagram just by clicking the diagram button and by setting some options in the wizard. This diagram depicting the distribution of various types of video file formats across Web servers is shown in Figure 9.7.

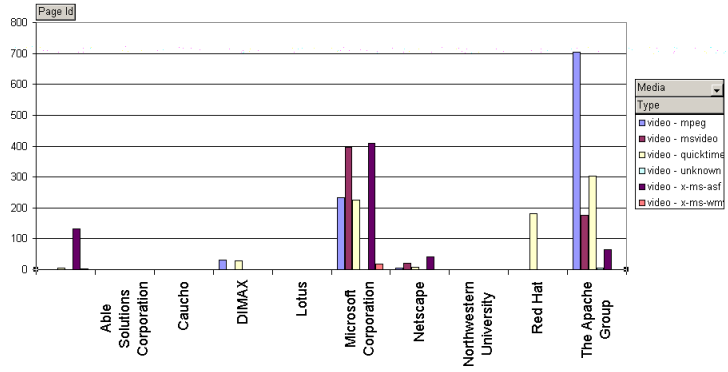


Figure 9.7: Distribution of video file types across Web servers

Several interesting aspects can be discovered when analyzing the distribution of file types across the different types of Web servers. General known tendencies, like the dominance of the PDF format over the previously very important Postscript file format for document exchange can be verified this way, as depicted in Figure 9.8.

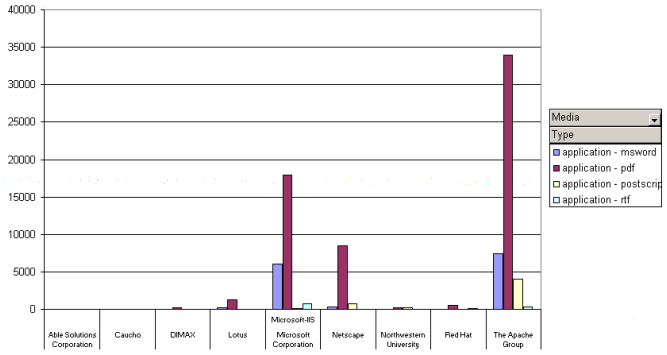


Figure 9.8: Distribution of document file types across Web servers

Similar characteristics can be detected when analyzing image file type distributions across different server types as depicted in Figure 9.9. Here we find an almost exclusive presence of the *png* file type on Apache servers, whereas more than 60% of all *bmp* files are to be found on MS IIS servers. However, when we take a look at the absolute distributions, we find that the *png* file format still

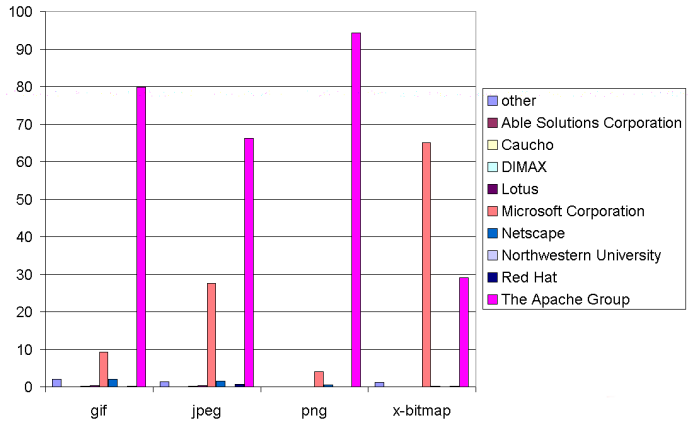


Figure 9.9: Relative distribution of image file types across Web servers

plays a neglectable role at this time, with a clear dominance of *jpeg*, followed by *gif* images.

9.2 Distribution of Web servers over the counties in Austria

Figure 9.10 represents a distribution graph of the domains in Austria, showing that most of the Web servers are located in the capital Vienna. If we make another slice by restricting the IP addresses to class A IPs only, the difference is even more obvious. Although this fact is not really surprising, the magnitude of the difference between the metropolis and the rest of Austria still is astounding, especially when we consider that just less then a quarter of the population lives in Vienna. More precisely, our analysis reveals that 66% of the Web-hosts are registered in Vienna, followed by Upper Austria with 9% and Styria with 6%. The distribution of the Web hosts in these other counties are comparable to the distribution of the population. This points towards the much-discussed issue of the "metropolitan media Internet".

However, care must be taken with respect to the information represented by the geographical domain, which reflects the location of the owner of a certain IP segment, rather than the actual location and area serviced by a specific server.

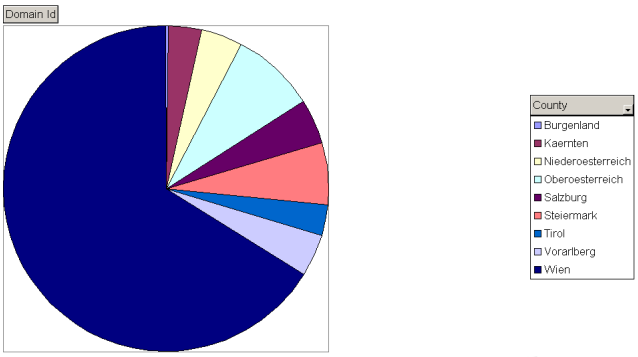


Figure 9.10: Distribution of Web hosts in Austria

As many nation-wide operating ISPs are based in Vienna, and thus have their address block registered there, the actual saturation and distribution of Internet services differs from the impression provided by this analysis. A combination with other means of location or geographical coverage determination should be incorporated to cover these issues, such as content-based coverage identification.

A drill-down onto the sub-domains provides a different view of the national distribution, where, for example, the academic and commercial nets are at least somewhat more evenly dispersed among the counties, whereas governmental Web sites as well as organizational sites are less wide-spread. Furthermore, we may not forget to take into account the "foreign" hosts, i.e. hosts registered in Austria, but registered under foreign domains, which currently amount to more than 6.800 individual domains (or close to 9.000 if alias names of servers are considered independently). These are not assigned to any of the *.at* sub-domains. such as e.g., some Austrian University Institutes that have their Web space located directly under the top-level *.edu* domain.

9.3 Distribution of Web servers across domains

While the distribution of the different Web servers used on the Web is one of the most frequently analyzed facts, and thus in itself does not reveal any surprising results, the application of DWH technology allows us to more flexibly view the various facets of this subject. We came across 35 different types of servers or

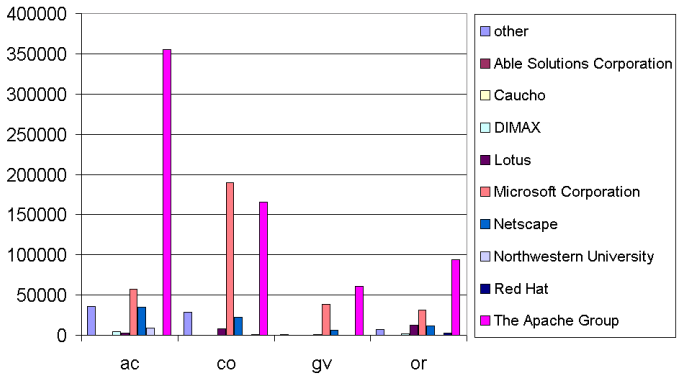


Figure 9.11: Distribution of Web servers across domains

server producers, in a total of about 300 different versions, but the most common ones are the APACHE and the IIS server, followed by the Netscape-Enterprise server. For a selection of the various types encountered, see Table 9.2.

By drilling-down we can take a look at the distribution of Web servers at the first sub-domain level. Figure 9.11 depicts the resulting distribution focusing on the most prominent types of Web servers. The general trends in market shares remain more or less unchanged, with probably a slightly stronger dominance of the Apache Web server in the academic domain. However, an interesting characteristic is represented by the presence of the WN Web server from Northwestern University, an open-source Web server that is used exclusively in the academic domain. This anomaly becomes even more obvious when we take a look at the relative distributions, depicted in Figure 9.12. Here the dominance of Apache in all but the governmental domains is clearly visible.

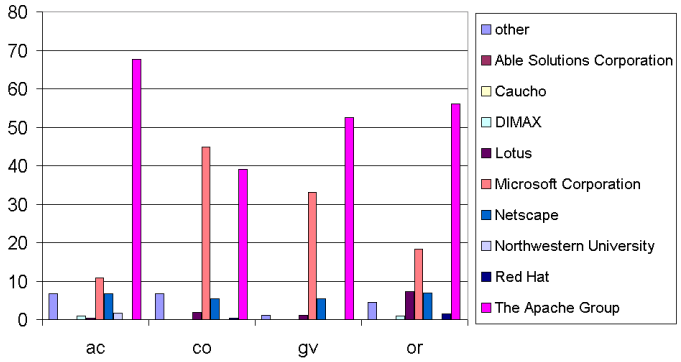


Figure 9.12: Relative distribution of Web servers across domains

Producer	**# Versions**	**# Ocurrences**
4D	13	361
Able Solutions	1	10
Apple	5	24
Caucho	5	19
DIMAX	4	68
IBM	14	78
Lotus	2	506
Microsoft Corporation	7	20947
NCSA	4	48
Netscape	26	1509
Northwestern Univ.	1	63
Novell	3	138
RapidSite	2	438
Red Hat	6	297
Roxen	2	102
The Apache Group	52	47383

Table 9.2: Selection of server types and versions encountered

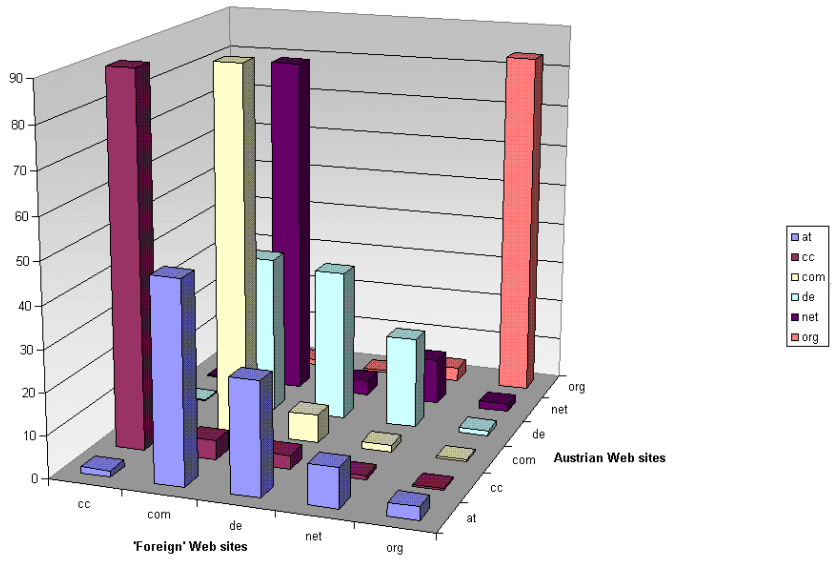


Figure 9.13: Austrian links pointing into other top-level domains

9.4 Link Analysis

In this section I will show some interesting results regarding the HTML links of the Web sites in the Austrian Web space. In this area of analysis there is a big space for improvement. We have to consider that our results are based on just the HTML links. There are a lot of other possibilities to point from one Web page to another like, for example, using dynamic HTML or flash just to mention two of them.

9.4.1 Austrian links pointing into other top-level domains

In Figure 9.13 you can see the relative distribution of the top-level domains where the links from the Austrian Web sites point to. You have to consider that the target top-level domains include just Web sites, which are not in the Austrian Web space ('foreign' Web sites). This is the reason why there is no '.at' top-level domain at the target domains shown on the x-axis in Figure 9.13. Some interesting points can be read out of the graphic. Sites from the '.at' domain are

mainly pointing to the '.com' and '.de' domain. When we have a closer look at the links from the '.org' domain you will see that nearly all of them are pointing into the same domain '.org'. Also the links of the top-level domain '.cc' and '.com' are pointing mainly into the same domains '.cc' and '.com'. That is not too surprising but when we have a look at the top-level domain '.de' we can see that more of these links are pointing to the other top-level domain '.com'. And even more apparently is this phenomenon at the top-level domain '.net' where far most of the links are pointing to '.com'. Let us have a look at the '.net' in more detail. We drill through the dimension to see which Web sites are linking to the '.com' domain. As we can see in Figure 9.14 it is just one Web site namely `tucows-servers.austro.net`, which causes this anomaly. We also drilled down the dimension *AllLinks* to see where exactly these links are pointing to. It turns out that most of these links are pointing to the respective Web site `www.tucows.com` a provider of internet services in the '.com' domain.

Page Id	Toplevel D	Sub Domain	Sub Sub Domain	Sub Sub Domain	Sub Sub Sub										
	at	cc	com	de	net										
					#NZ										
					a										
					a1	austrianinline	austria-perf	austriaweb	austriawine	austro					
Link Toplev										fbg	minoriten	trenner	tucows-server	users	www
cc	321657	429787	5358	8	15			6	7				3	1	
ch	483473	89	944	8	4		1	59	14				4	3	
com	10271134	22009	541190	1369	59	371	1	657	2476	49		7	87417	184	9
de	5767734	16263	41593	1308	99	24	41	912	117	43	9	2	29	76	
edu	120657	42	1166					38					4	2	
localhost	172897		1												
net	2050678	4835	10558	788	6641	16	2	421	1816		30	2	478	46	
org	661016	2179	3342	47	7		2	100	43			1	56	13	
uk	266926	194	1052	2		2		16					63	2	
Grand Total *	21506271	482313	613119	3639	6834	415	47	2355	4635	93	41	12	88327	351	9

Figure 9.14: Top-level domain '.net' pointing to '.com'

In Table 9.3 you can see the total amount of links appearing in the different top-level domains of the Austrian Web space compared to the occurring different URLs.

9.4.2 Searching for relations between different sites in the Web space

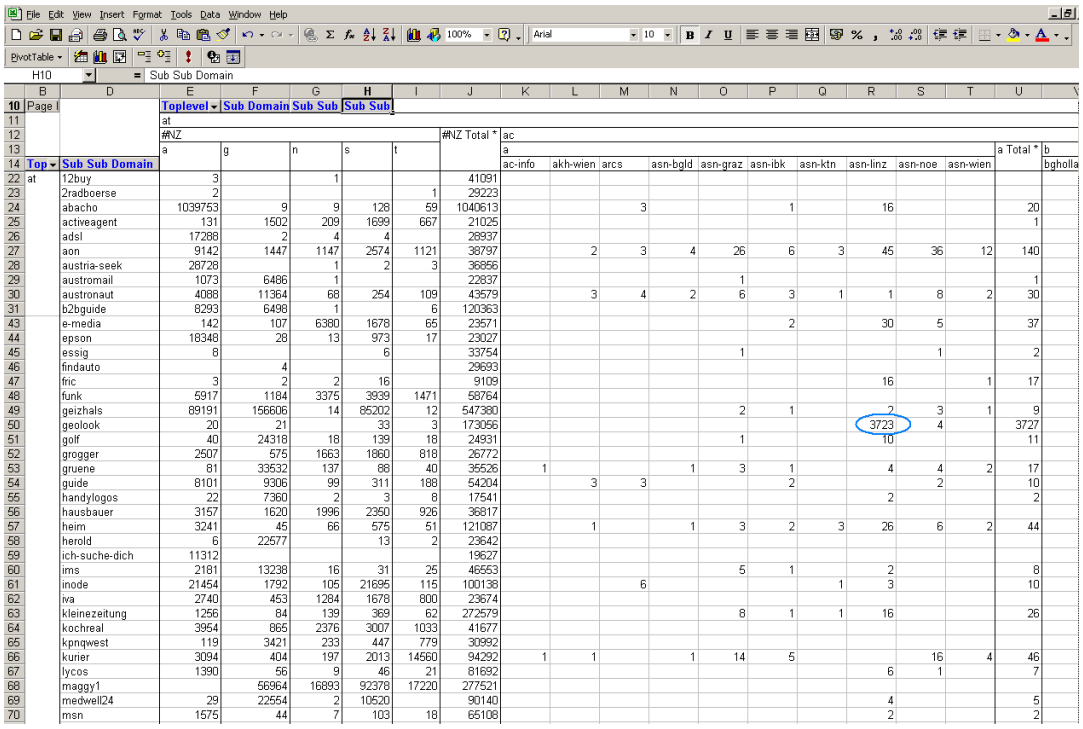
In this section we will have a look at the relations between different Web sites in the space of the Austrian Web. For this analysis we choose the cube containing

Top-level domains	Number of URLs	Number of links
.at	12 600 039	38 590 804
.cc	177 093	518 096
.com	160 457	923 988
.net	92 122	266 967
.org	21 994	463 227
.de	3 114	4 519
others	8 675	14 588
Total	13 063 494	40 782 189

Table 9.3: Number of URLs compared to the number of links

the dimensions *Domains* for the Web sites, the dimension *Linked_AOLA* for the linked Web sites in the Austrian Web space, the dimension *OS* for the operating systems, and the dimension representing the Web servers. First, we export the data to Microsoft Excel as described in Section 8.2. In the spreadsheet we choose the dimensions *Linked_AOLA* for the y-axis and the *Domains* for the x-axis. The center of our data grid is filled with the number of links pointing from the Web sites shown at the x-axis to the Web sites at the y-axis. As we can see in Figure 9.15 there is a conspicuous characteristic at the Web site `www.asn-linz.ac.at`. Most of the links from this Web site are pointing to another Web site namely `www.geolook.at`.

We will have a closer look at the domain `www.asn-linz.ac.at` of the 'Austrian School Network Linz'. It is a portal about the educational offers in Austria. There is a link to all the schools and universities in Austria. In Table 9.4 we can see the domains where most of the links at this domain are going to. As we already detected, the domain (`www.geolook.at`) is mostly linked by this portal. This site provides an integration of interactive, dynamic maps of Austria. When we looked at the portal of 'Austrian School Network Linz', we found out that there is the possibility to look up the according map, for each school and university in Austria. That is why the Web site `www.asn-linz.at` and the site `www.geolook.at` has such a relationship comparable with a hub-



Top	Sub Sub Domain	a	g	n	s	t	#NZ Total *	ac-info	akh-wien	arcs	asn-bgld	asn-graz	asn-ibk	asn-ktn	asn-linz	asn-noe	asn-wien	a Total *	b bgholla
at	12buy	3		1			41091												
	2radboerse	2				1	29223												
	abacho	1039753	9	9	128	59	1040613			3			1		16			20	
	activeagent	131	1502	209	1699	667	21025											1	
	adsl	17288	2	4	4		28937												
	aon	9142	1447	1147	2574	1121	38797		2	3	4	26	6	3	45	36	12	140	
	austria-seek	28728		1	2	3	36856												
	austromail	1073	6486	1			22837					1						1	
	austronaut	4088	11364	68	254	109	43579		3	4	2	6	3	1	1	8	2	30	
	b2bguide	8293	6498	1		6	120363												
	e-media	142	107	6380	1678	65	23571						2		30	5		37	
	epson	18348	28	13	973	17	23027												
	essig	8			6		33754					1				1		2	
	findauto		4				29693												
	fric	3	2	2	16		9109								16		1	17	
	funk	5917	1184	3375	3939	1471	58764												
	geizhals	89191	156606	14	85202	12	547380					2	1		2	3	1	9	
	geolook	20	21		33	3	173056								3723	4		3727	
	golf	40	24318	18	139	18	24931					1			10			11	
	grogger	2507	575	1663	1860	818	26772												
	gruene	81	33532	137	88	40	35526	1			1	3	1		4	4	2	17	
	guide	8101	9306	99	311	188	54204		3	3			2			2		10	
	handylogos	22	7360	2	3	8	17541								2			2	
	hausbauer	3157	1620	1996	2350	926	36817												
	heim	3241	45	66	575	51	121087		1		1	3	2	3	26	6	2	44	
	herold	6	22577		13	2	23642												
	ich-suche-dich	11312					19627												
	ims	2181	13238	16	31	25	46553					5	1		2			8	
	inode	21454	1792	105	21695	115	100138			6				1	3			10	
	iva	2740	453	1284	1678	800	23674												
	kleinezeitung	1256	84	139	369	62	272579					8	1	1	16			26	
	kochreal	3954	865	2376	3007	1033	41677												
	kpnqwest	119	3421	233	447	779	30992												
	kurier	3094	404	197	2013	14560	94292	1	1		1	14	5			16	4	46	
	lycos	1390	56	9	46	21	81692								6	1		7	
	maggy1		56964	16893	92378	17220	277521												
	medwell24	29	22554	2	10520		90140								4			5	
	msn	1575	44	7	103	18	65108								2			2	

Figure 9.15: Table showing the number of links

	www.asn-linz.ac.at
www.geolook.at	3 723
uibk.ac.at	315
univie.ac.at	251
kfunigraz.ac.at	173
Total number of links	38 404

Table 9.4: Relation of two domains

authority relation with the domain `www.asn-linz.at` as the hub and the domain `www.geolook.at` as the authority. Further sites linked by this domain are very often sites of the different universities in Austria.

9.4.3 Some Link statistics

In this section some interesting high score rankings of links are shown. First, we have a look at the domains in the Austrian Web space containing most links. Second, the domains of the Austrian Web space, which are linked by other sites are presented (most backlinks).

Domains containing most links In Table 9.5 the top ten domains in the Austrian Web space containing most links are shown. The top domain `lion.cc` is a big shopping portal in Austria where you can buy books, videos, etc. The second domain `univie.ac.at` presents the main university in Vienna, the capital of Austria. `Tu-graz.ac.at` and `tuwien.ac.at` are the domains of the technical universities of Vienna and Graz, the second biggest city of Austria. The domain `wu-wien.ac.at` represents the university of economy in Vienna. 'Tiscover' is a tourist agency portal of Austria providing every kind of tourist information. You also have the possibility to book online at this portal. The domain `uni-linz.ac.at` represents the university in Linz (Upper Austria). 'Austronaut' is an Austrian search engine. The domain `oberoesterreich.com` represents an info portal containing mainly Austrian news. The domain `austro.net` represents 'CyberTron', a big Austrian telecommunication company.

Domains	Number of links
lion.cc	476 767
univie.ac.at	344 180
tu-graz.ac.at	249 832
tuwien.ac.at	244 642
wu-wien.ac.at	181 285
tiscover.com	176 028
uni-linz.ac.at	152 593
austronaut.com	104 216
oberoesterreich.com	98 276
austro.net	89414

Table 9.5: Top 10 sub-level domains containing the most links

When we drill down one more level, we can see that the number of links are sometimes distributed over several third-level domains, as it is shown in Figure 9.16 for the domain `lion.cc`.

In Table 9.6 you can see the top ten domains with the most links drilled down one more level. The top domain `austria.indymedia.org` is part of the portal called 'Independent Media Center'. It is a network of collectively media outlets funded in the USA. Please refer to the explanation above for the other domains.

Domains containing most backlinks In Table 9.7 the top ten domains in the Austrian Web space with most backlinks are shown. As already described, backlinks are links pointing to these sites. The domains with the most backlinks are again the domains of the biggest universities in Austria. The domain `rainbow.or.at` represents a gay and lesbian communication forum. The domains `noe.gv.at`, and `stmk.gv.at` are portals of the state governments of the counties 'Lower Austria', and 'Styria'. The domain `magwien.gv.at` represents the magistrate of Vienna.

					MeasuresLevel
- Toplevel Domain	- Sub Domain	- Sub Sub Domain Group...	- Sub Sub Domain	+ Sub Sub Sub Domain	Page Id
- cc	- #NZ	- l	- lion	+ feedback	10.255
				+ film	17.061
				+ forum	1
				+ foto	660
				+ fun	653
				+ future	62
				+ game	11.790
				+ games	9.256
				+ gourmet	45
				+ hilfe	19.725
				+ homepage	44.861
				+ hpuser	476
				+ info	8.881
				+ jazz	94
				+ jobs	12.477
				+ kommunikation	13.995
				+ lesen	43
				+ life	437
				+ magazine	10.476
				+ mp3	26.631
				+ mp3radio	
				+ music	8.328
				+ musik	14.633
				+ my	11.392
				+ mylion	10.774
				+ nachrichten	15.179
				+ news	581
				+ partnerprogramm	243
				+ partnerzone	1.384
				+ privacy	6.992
				+ register	
				+ registrierung	316
				+ shop	10.746
				+ shops	20.018
				+ sms	115
				+ software	8.411
				+ standard	7.181
				+ start	4.862
				+ thema	115
				+ themen	769
				+ tv	9
				+ verlag	84
				+ video	8.509
				+ webinsel	1.418
				+ webland	22.527
				+ webmail	
				+ webplanet	1.205
				+ websport	2.629
				+ webstadt	824
				+ wetter	30.186

Figure 9.16: Drilldown of 'lion.cc'

Domains	Number of links
austria.indymedia.org	441 150
linuxberg.univie.ac.at	137 406
tucows.tu-graz.ac.at	134 235
www.austronaut.com	104 148
cms.tiscover.com	82 088
linuxberg.tu-graz.ac.at	78 056
cms0.tiscover.com	74 029
finder.oberoesterreich.com	62 726
tucows.univie.ac.at	52 692
homepage.lion.cc	44 861

Table 9.6: Top 10 sub-sub-level domains containing the most links

Domains	Number of backlinks
wu-wien.ac.at	241 811
tuwien.ac.at	152 436
tu-graz.ac.at	147 662
univie.ac.at	79 135
uni-linz	78 160
rainbow.or.at	73 402
boku.ac.at	68 884
noe.gv.at	62 058
stmk.gv.at	44 321
magwien.gv.at	42 534

Table 9.7: Top 10 domains containing the most backlinks

Recapitulating this section, we see that the biggest link communities can be found among educational Web sites (mainly universities) followed by governmental sites, which are mainly linked by other sites. Additionally a few big commercial Web sites were detected, representing a dense network of links.

9.5 Conclusions

In this chapter I picked out just a few of the many possible analyses. Our project takes advantage of possibilities of a modern Data Warehouse to provide flexible and interactive analyses of the various characteristics of the Web.

The results, summaries, and statistics correspond to groups-bys on different dimensions, multiple abstraction levels, and their arbitrary combinations. Dicing, slicing, and drilling can be performed on the data cubes to examine different types of requested statistics.

Section 9.1 shows a sample analyzing process step by step, by providing the distribution of file-types over the different Web servers. In Section 9.2 we took a look at the distribution of Web servers over the counties in Austria, whereas in Section 9.3 I focused on the distribution of Web servers across the domains. In Section 9.4 I picked out some sample results of the analysis process of the link structure of the Web.

Chapter 10

Hardware and Software used

The hardware and the respective software used in this project are as follows:

- *Processor*: AMD Athlon MMX 900MHZ
- *Memory*: 768 MB RAM
- *Operating system*: Microsoft Windows 2000
- *Database*: Microsoft SQL Server 2000
- *OLAP tool*: Microsoft Analysis Manager
- *tool creating the charts*: Microsoft Excel

The software used seems to be very good for the requirements of our project. Due to the fact that we are using only Microsoft software, the communication between each software is very easy to handle. The database is designed for this amount of data, we used and even more. As described in Section 8.1 the import of the data into the Excel program is very easy to handle too.

The feeding process of the database took a long time (several weeks). Calculating the aggregation by the OLAP tool, processing the dimensions, and even drilling up and down takes a while. In my opinion the size limit of the Data Warehouse is nearly reached to be able to work with reasonable response times. When we are feeding another snapshot of the Web, we will have to upgrade the hardware to handle the amount of data.

Chapter 11

Conclusions

The World-Wide Web is continuously growing and is already the biggest data repository ever built. An important challenge is the collection of structural information of the Web to allow us to analyze and understand it in its full complexity, benefiting from the wealth of information provided by it, which goes beyond mere content analysis. While the improvement of Web search results may be facilitated by the collection and integration of additional information such as link structure analysis, by far more fascinating insights into the Web and its evolution will become possible. These include the evolution and maturation of technologies employed, analysis of market shares, but also, from a preservation perspective, technologies and efforts required to preserve the diversity of information representation. While several of these issues have been addressed in various projects employing special purpose tools, the integration of the wealth of data associated with the Web into a Data Warehouse opens the doors for more flexible analysis of this medium.

In this thesis I have presented the *Austrian On-Line Archive Processing (AOLAP)* project. Web data taken from crawls of the Austrian Web space as part of the AOLA initiative has been combined with information obtained from additional sources, such as the WHOIS database. The resulting Data Warehouse allows for flexible analysis and exploration of the selected Web space, and – via the integration of data from subsequent crawls – its evolution over time.

One of the biggest advantages of our project is the possibility to modify and

change the analysis environment to provide a solution for several kinds of tasks. In Chapter 9 I provided just a few examples of the many possible analysis.

11.1 Open work and further improvements

Certainly much effort can be put into the project for future improvements. Further types of information can be extracted from the pages, integrating e.g. automatic language detection methods, covering in larger detail additional technological information, such as the usage of cookies, embedded java applets, flash plug-ins, encryption, etc., in order to be able to incorporate future technologies. Furthermore, the addition of a content-based dimension is being considered. As part of these expansions flexible interfaces to modify/increase the number and type of technologies to be scanned for in the data, will be analyzed. Due to this new content information it will be possible to distinguish between different types of links. For example, we want to be able to recognize links between pages that are related because of common administrative control or advertising, paid links. For more information about this issue, please refer to the paper *Recognizing Nepotistic Links on the Web* [47]. Furthermore, the application of specific data mining techniques for specific problem domains will be studied in greater detail. Further crawls of the Web are also required to make more detailed analysis over the Austrian Web space with respect to the time-line.

Appendix A

List of different Web Server

Producer	Server Product	# Versions	# Ocurrences
4D	WebSTAR	1.0	4
4D	WebSTAR	1.2.4	1
4D	WebSTAR	1.3.2	3
4D	WebSTAR	2.1	57
4D	WebSTAR	2.1.1	12
4D	WebSTAR	3.0	20
4D	WebSTAR	3.0.1	5
4D	WebSTAR	3.0.2	24
4D	WebSTAR	4.0	45
4D	WebSTAR	4.1	11
4D	WebSTAR	4.3	78
4D	WebSTAR	4.4	96
4D	ACI-4D	6.56	5
Able Solutions	Commerce-Builder	2.0	10
ACME	thttpd	2.05	3
ACME	thttpd	2.19	3
Alibaba	Alibaba	2.0	4
America Online	AOLserver	3.2	1
Apple	AppleShareIP	6.0.0	2
Apple	AppleShareIP	6.3.0	2
Apple	AppleShareIP	6.3.1	8
Apple	AppleShareIP	6.3.2	12
Apple	AppleShareIP	6.3.3	7
Blueworld	Lasso	3.0	5
Blueworld	Lasso	3.5	2
Blueworld	Lasso	3.6.5	23
BorderWare	BorderWare	2.2	3
Caucho	Resin	1.2.7	10
Caucho	Resin	1.3.b1	2
Caucho	Resin	2.0.2	4

to be continued...

Producer	Server Product	# Versions	# Ocurrences
Caucho	Resin	2.0.4	1
Caucho	Resin	2.0.5	2
Cern	CERN	3.0	2
Cern	CERN	3.0A	1
Cern	CERN	3.0pre6	2
Cern	CERN	3.0pre6vms3	1
Deerfield.com	WebSitePro	1.1g	1
Deerfield.com	WebSitePro	1.1h	2
Deerfield.com	WebSitePro	2.0.36	7
Deerfield.com	WebSitePro	2.3.18	3
Deerfield.com	WebSitePro	2.3.7	4
Deerfield.com	WebSitePro	2.4.9	55
Deerfield.com	WebSitePro	2.5.4	4
Deerfield.com	WebSitePro	2.5.8	23
Deerfield.com	WebSitePro	3.0.37	8
DIMAX	Hyperwave-Info-Server	2.5	1
DIMAX	Hyperwave-Info-Server	4.1	6
DIMAX	Hyperwave-Info-Server	5.1.1	5
DIMAX	Hyperwave-Info-Server	5.5	56
GNU Software	icecast	1.3.10	2
IBM	Domino-Go-Webserver	4.6.1	2
IBM	Domino-Go-Webserver	4.6.2.5	8
IBM	Domino-Go-Webserver	4.6.2.6	7
IBM	GoServe	2.50	4
IBM	IBM-HTTP-Server	1.0	21
IBM	IBM-HTTP-Server	1.3.12	1
IBM	IBM-HTTP-Server	1.3.12.1	9
IBM	IBM-HTTP-Server	1.3.12.2	9
IBM	IBM-HTTP-Server	1.3.12.3	3
IBM	IBM-HTTP-Server	1.3.19	2
IBM	IBM-HTTP-Server	1.3.3.1	2
IBM	IBM-HTTP-Server	1.3.6	4
IBM	IBM-HTTP-Server	1.3.6.1	5
IBM	IBM-HTTP-Server	1.3.6.2	1
iMatix	Xitami	Pro	128
Ironflare	Orion	1.4.4	3
Ironflare	Orion	1.4.5	2
Ironflare	Orion	1.4.7	1
Ironflare	Orion	1.5.1	4
Ironflare	Orion	1.5.2	6
Lotus	Lotus-Domino	(4.5 - 5.0)	506
Microsoft	Microsoft-IIS	2.0	9
Microsoft	Microsoft-IIS	3.0	305
Microsoft	Microsoft-IIS	4.0	11262
Microsoft	Microsoft-IIS	5.0	9356
Microsoft	Microsoft-IIS	6.0	2
Microsoft	Microsoft-PWS	2.0	9

to be continued...

Producer	Server Product	# Versions	# Ocurrences
Microsoft	Microsoft-PWS	3.0	4
NCSA	NCSA	1.4.1	3
NCSA	NCSA	1.4.2	11
NCSA	NCSA	1.5	2
NCSA	NCSA	1.5.2	32
NetLink	NetLink	4D	2
Netscape	Netscape-Commerce	1.1	1
Netscape	Netscape-Communications	1.1	3
Netscape	Netscape-Communications	1.12	3
Netscape	Netscape-Enterprise	2.01	20
Netscape	Netscape-Enterprise	2.01c	3
Netscape	Netscape-Enterprise	2.01d	6
Netscape	Netscape-Enterprise	2.0a	10
Netscape	Netscape-Enterprise	2.0d	4
Netscape	Netscape-Enterprise	3.0	3
Netscape	Netscape-Enterprise	3.0C	1
Netscape	Netscape-Enterprise	3.0F	1
Netscape	Netscape-Enterprise	3.5.1	35
Netscape	Netscape-Enterprise	3.5.1C	73
Netscape	Netscape-Enterprise	3.5.1G	59
Netscape	Netscape-Enterprise	3.5.1I	6
Netscape	Netscape-Enterprise	3.5-For-NetWare	16
Netscape	Netscape-Enterprise	3.6	276
Netscape	Netscape-Enterprise	4.0	31
Netscape	Netscape-Enterprise	4.1	740
Netscape	Netscape-Enterprise	6.0	5
Netscape	Netscape-FastTrack	2.01	20
Netscape	Netscape-FastTrack	2.0a	10
Netscape	Netscape-FastTrack	2.0c	40
Netscape	Netscape-FastTrack	3.01B	4
Netscape	Netscape-FastTrack	3.02	18
Netscape	Netscape-FastTrack	3.5-For-NetWare	4
Northwestern Univ.	WN	(2.0 - 2.2)	63
Novell	NetWare-Enterprise	5.1	117
Novell	Novell-HTTP-Server	2.51R1	2
Novell	Novell-HTTP-Server	3.1R1	19
Ohio State Univ.	OSU	1.9a	1
Ohio State Univ.	OSU	1.9c	1
Ohio State Univ.	OSU	3.3b	3
Ohio State Univ.	OSU	3.9	2
Ohio State Univ.	OSU	3.9a	4
Omnicron Technologies	OmniHTTPd	2.06	8
Omnicron Technologies	OmniHTTPd	2.08	16
OpenSA Project	OpenSA	XXX	6
Oracle	Oracle Web Listener	(1.2 - 4.0)	57
RapidSite	Rapidsite	1.3.14	437
RapidSite	Rapidsite	1.3.4	1

to be continued...

Producer	Server Product	# Versions	# Ocurrences
Red Hat	Stronghold	2.0.1	4
Red Hat	Stronghold	2.0b1	26
Red Hat	Stronghold	2.2	175
Red Hat	Stronghold	2.4.1	1
Red Hat	Stronghold	2.4.2	42
Red Hat	Stronghold	3.0	49
Roxen	Roxen	(1.3 - 2.2)	53
Roxen	Roxen Challenger	(1.2 - 1.3)	49
Spinner	Spinner	1.0b12	3
Sun Microsystems	JavaWebServer	1.1.3	5
Sun Microsystems	JavaWebServer	2.0	3
The Apache Group	Apache	1.0.3	1
The Apache Group	Apache	1.0.5	1
The Apache Group	Apache	1.1.1	26
The Apache Group	Apache	1.1.3	12
The Apache Group	Apache	1.2.0	11
The Apache Group	Apache	1.2.1	29
The Apache Group	Apache	1.2.4	65
The Apache Group	Apache	1.2.5	87
The Apache Group	Apache	1.2.6	752
The Apache Group	Apache	1.2.6.6	2
The Apache Group	Apache	1.2.7	2
The Apache Group	Apache	1.2b0	2
The Apache Group	Apache	1.2b10	8
The Apache Group	Apache	1.2b6	9
The Apache Group	Apache	1.2b7	4
The Apache Group	Apache	1.2b8	4
The Apache Group	Apache	1.3.0	182
The Apache Group	Apache	1.3.1	89
The Apache Group	Apache	1.3.1.1	572
The Apache Group	Apache	1.3.11	768
The Apache Group	Apache	1.3.12	11705
The Apache Group	Apache	1.3.13	2
The Apache Group	Apache	1.3.14	5403
The Apache Group	Apache	1.3.17	1010
The Apache Group	Apache	1.3.19	6021
The Apache Group	Apache	1.3.2	55
The Apache Group	Apache	1.3.20	6278
The Apache Group	Apache	1.3.20a	3
The Apache Group	Apache	1.3.22	3741
The Apache Group	Apache	1.3.23	460
The Apache Group	Apache	1.3.24	21
The Apache Group	Apache	1.3.3	938
The Apache Group	Apache	1.3.4	399
The Apache Group	Apache	1.3.5	1
The Apache Group	Apache	1.3.6	3132
The Apache Group	Apache	1.3.9	5407

to be continued...

Producer	Server Product	# Versions	# Ocurrences
The Apache Group	Apache	1.3a1	2
The Apache Group	Apache	1.3b3	8
The Apache Group	Apache	1.3b5	3
The Apache Group	Apache	2.0.18	2
The Apache Group	Apache	2.0.28	1
The Apache Group	Apache	2.0a3	1
The Apache Group	Apache	df-exts	18
The Apache Group	AdvancedExtranet	1.3.12	11
The Apache Group	AdvancedExtranet	1.3.14	20
The Apache Group	AdvancedExtranet	1.3.19	17
The Apache Group	AdvancedExtranet	1.3.20	17
The Apache Group	AdvancedExtranet	1.3.22	13
The Apache Group	AdvancedExtranet	1.3.23	2
The Apache Group	ApachePro	1.3.14-11	1
The Apache Group	mod-perl	1.15	1
The Apache Group	mod-perl	1.18	59
UIC	FirstClass	5.5	15
UIC	FirstClass	6.0	15
Zeus Technology	Zeus	3.3	73
Zeus Technology	Zeus	3.4	9
Zeus Technology	Zeus	4.0	6
Zope Community	Zope	Zope	48

Table A.1: Types of Servers

Bibliography

[1] A. Ardö and S. Lundberg. A regional distributed www search and indexing service - the DESIRE way. In *Proceedings of the 7. WWInternational World Wide WebW Conference (WWW7)*, Brisbane, April 14-18 1998. `http://www7.scu.edu.au/programme/fullpapers/1900/com1900.htm`.

[2] A. Arvidson, K. Persson, and J. Mannerheim. The Kulturarw3 project - The Royal Swedish Web Archiw3e - An example of completecollection of web pages. In *Proceedings of the 66th IFLA Council and General Conference*, Jerusalem, Israel, August 13-18 2000.
`http://www.ifla.org/IV/ifla66/papers/154-157e.htm`.

[3] Nordic Web Archive `http://nwa.nb.no`

[4] S.B. Bhowmick, N.W. Keong, and S.K. Madria. Web schemas in WHOWEDA. In *Proceedings of the ACM 3rd International Workshop on Data Warehousing and OLAP*, Washington, DC, November 10 2000. ACM.

[5] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tompkins, and J. Wiener. Graph structure in the web. In *Proceedings of the 9th World Wide Web Conference (WWW9)*, Amsterdam, Netherlands, May 15-19 2000.
`http://www9.org/w9cdrom/160/160.html`.

[6] J. Ding, L. Gravano, and N. Shivakumar. Computing geographical scopes of web resources. In *Proceedings of the 26th International Conference on Very Large Databases, VLDB 2000*, pages 545–556, Cairo, Egypt, September 10-14 2000.

[7] J. Hakala. Collecting and preserving the web: Developing and testing the NEDLIB harvester. *RLG DigiNews*, 5(2), April 15 2001. `http://www.rlg.org/preserv/diginews/diginews5-2.html#feature2`.

[8] J.M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[9] S.K. Madria, S.S. Bhowmick, W.K. Ng, and E.P. Lim. Research issues in web data mining. In *Proceedings of the 1st International Conference on Data Warehousing and Knowledge Discovery (DAWAK99)*, LNCS 1676, pages 303–312, Florence, Italy, August 30 - September 3 1999.

[10] M. Najok and J.L. Wiener. Breadth-first search crawling yields high-quality pages. In *Proceedings of the 10th World Wide Web Conference (WWW10)*, Hong-Kong, May 2-5 2001. ACM. `http://www10.org/cdrom/papers/208/`.

[11] A. Rauber and A. Aschenbrenner. Part of our culture is born digital - On efforts to preserve it for future generations. *TRANS. On-line Journal for Cultural Studies (Internet-Zeitschrift für Kulturwissenschaften)*, 10, July 2001. `http://www.inst.at/trans/10Nr/inhalt10.htm`.

[12] B. Mobasher, N. Jain, E. Han, and J. Srivastava. Web mining: Pattern discovery from world wide web transactions. *Technical Report TR 96-050, University of Minnesota, Dept. of Computer Science* Minneapolis, 1996.

[13] J. Pitkow and Krishna K. Bharat. Webviz: A tool for world-wide web access log analysis. *In First International WWW Conference* 1994.

[14] C. Dyreson. Using an incomplete data cube as a summary data sieve. *Bulletin of the IEEE Technical Committee on Data Engineering* pages 19–26, March 1997.

[15] C. M. Brown, B. B. Danzig, D. Hardy, U. Manber, and M. F. Schwartz. The harvest information discovery and access system. *In Proc. 2nd International World Wide Web Conference*, 1994.

[16] K. Hammond, R. Burke, C. Martin, and S. Lytinen. Faq-finder: A case-based approach to knowledge navigation. *In Working Notes of the AAAI Spring Symposium: Information Gathering from Heterogeneous, Distributed Environments.* AAAI Press, 1995.

[17] T. Kirk, A. Y. Levy, Y. Sagiv, and D. Srivastava. The information manifold. *In Working Notes of the AAAI Spring Symposium: Information Gathering from Heterogeneous, Distributed Environments.* AAAI Press, 1995.

[18] C. Kwok and D. Weld. Planning to gather information. *In Proc. 14th National Conference on AI*, 1996.

[19] E. Spertus. Parasite: mining structural information on the web. *In Proc. of 6th International World Wide Web Conference*, 1997.

[20] R. B. Doorenbos, O. Etzioni, and D. S. Weld. A scalable comparison shopping agent for the world wide web. *Technical Report 96-01-03, University of Washington, Dept. of Computer Science and Engineering*, 1996.

[21] M. Perkowitz and O. Etzioni. Category translation: learning to understand information on the internet. *In Proc. 15th International Joint Conference on AI* pages 930–936, Montral, Canada, 1995.

[22] W. B. Frakes and R. Baeza-Yates. Information Retrieval Data Structures and Algorithms. *Prentice Hall, Englewood Cliffs*, NJ, 1992.

[23] R. Weiss, B. Velez, M. A. Sheldon, C. Namprempre, P. Szilagyi, A. Duda, and D. K. Gifford. Hypursuit: a hierarchical network search engine that exploits content-link hpertexxt clustering. *In Hypertext'96: The Seventh ACM Conference on Hypertext*, 1996.

[24] R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell. Webwatcher: A learning apprentice for the world wide web. *In Proc. AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments.* 1995.

[25] K. A. Oostendorp, W. F. Punch, and R. W. Wiggins. A tool for individualizing the web. *In Proc. 2nd International World Wide Web Conference*, 1994.

[26] M. Pazzani, J. Muramatsu, and D. Billsus. Syskill & webert: Identifying interesting web sites. *In Proc. AAAI Spring Symposium on Machine Learning in Information Access*, Portland, Oregon, 1996.

[27] O. R. Zaiane and J. Han. Resource and knowledge discovery in global information systems: A preliminary design and experiment. *In Proc. of the First Int'l Conference on Knowledge Discovery and Data Mining* pages 331–336, Montreal, Quebec, 1995.

[28] I. Khosla, B. Kuhn, and N. Soparkar. Database search using informatiuon mining. *In Proc. of 1996 ACM-SIGMOD Int. Conf. on Management of Data*, 1996.

[29] R. King and M. Novak. Supporting information infrastructure for distributed, heterogeneous knowledge discovery. *In Proc. SIGMOD 96 Workshop on Research Issues on Data Mining and Knowledge Discovery*, Montreal, Canada, 1996.

[30] P. Merialdo P. Atzeni, G. Mecca. Semistructured and structured data in the web: Going back and forth. *In Proceedings of the Workshop on the Management of Semistructured Data (in conjunction with ACM SIGMOD)*, 1997.

[31] D. Konopnicki and O. Shmueli. W3qs: A query system for the world wide web. *In Proc. of the 21th VLDB Conference* pages 54–65, Zurich, 1995.

[32] Larry Page, Sergey Brin, R.Motwani, T.Winograd. The PageRank Citation Ranking: Bringing Order to the Web. *Stanford Digital Library Technologies Project* 1998.

[33] Dell Zhang, Yisheng Dong. An Efficient Algorithm to Rank Web Resources. *In Proceedings of the 9th International World Wide Web Conference, Amsterdam, Netherlands* May 2000.

[34] S. Leung, S. Perl, R. Stata, and J. Wiener. Towards web-scale web archeology. Research Report 174, Compaq Systems Research Center, Palo Alto, CA, September 10 2001.

[35] T. Werf-Davelaar. Long-term preservation of electronic publications: The NEDLIB project. *D-Lib Magazine*, 5(9), September 1999.

[36] J. Hirai, S. Raghavan, H. Garcia-Molina, and A. Paepcke. Webbase: A repositoru of web pages. In *Proc of the 9th International World Wide Web Conf (WWW9)*, Amsterdam, The Netherlands, May 15-19 2000. Elsevir Science.

[37] R. Kimball. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, 2 edition, 2002.

[38] T. Pedersen and C. Jensen. Multidimensional database technology. *IEEE Computer*, 34(12):40–46, December 2001.

[39] R. Bruckner and A. Tjoa. Managing time consistency for active data warehouse environments. In *Proceedings of the Third International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2001)*, LNCS 2114, pages 254–263, Munich, Germany, September 2001. Springer. `http://link.springer.de/link/service/series/0558/papers/2114/21140219.pdf`.

[40] M. Spiliopoulou, L.C. Faulstich. WUM:A Web Utilization Miner. In *EDBT Workshop WebDB'98*, Valencia, Spain, Mar. 1998. `http://wum.wiwi.hu-berlin.de/`.

[41] The Internet Archive: Building an Internet Library. `http://www.archive.org`

[42] F. Heylighen. Mining Associative Meanings from the Web: from word disambiguation to the global brain In *Proceedings of Trends in Special Language and Language Technology*, (Standaard Publishers, Brussels, 2001)

[43] update software AG ProspectMiner
`http://www.update.com/products/pm_en.html`

[44] IANA (Internet Assigned Numbers Authority) The IANA provide monitoring and coordination of Internet IP addresses and assignment. `http://www.iana.org`

[45] N. Pendse, R. Creeth. OLAP Report project
`http://www.olapreport.com`

[46] The file extension source
`http://filext.com`

[47] B.Davison. Recognizing Nepotistic Links on the Web Presented at the*AAAI-2000 Workshop on Artificial Intelligence for Web Search*, Austin, TX, July 30, 2000, and published in Artificial Intelligence for Web Search, Technical Report WS-00-01, pp. 23-28, AAAI Press.