Chapter 4

Introduction to Online Miner

Online Miner is an add-on software component that can inspect traffic at a deeper level than any other web-mining tool does. It is a software component that can be hosted online and can inspect the data before it allows to the web-mining repository. This tool starts its activities of gathering, filtering and categorization of data when the user moves or clicks the mouse button or key in the data into the web pages. This tool provides the transparency to transactional analysis on user behavior. Tool features are listed below.

Features

♦ Data Filters
♦ Rule set
♦ Live data
♦ Easy Management
♦ Flexibility
♦ Performance
♦ Individual focus
♦ Security
♦ Transformations through keywords
♦ Separation of data from presentation
♦ Easy Analysis/accurate reports
4.1. Online Miner Features

4.1.1. Data Filters

Web data being transmitted over a network channel on the Internet with excessive amount of data causes data processing problems, which include selectively choosing useful information to be retained for various data applications. Data filter is defined as a filter, which Captures the useful data and ignores the irrelevant data.

This tool provides an advanced technique to filter the data, since the filters are dynamically configured, providing a facility where the data can be deposited for storage in their respective categories. It facilitates the user (web site administrators) to add or modify the filters as per their requirement. These entire filter settings are stored in two tables known as Filter Accept and Filter Reject.

With the help of these filters the data collected is more accurate since the administrator deposits the data collected from the required page directly in to the arranged category.

4.1.1.1 Working with Filters

Example:

Home Accessories Company wants to place furniture ads on website integrated with Online Miner tool, web site administrator can configure the settings in the tool in such a way that they get all the transactions related to furniture. Then, basing on the given configuration the tool gets all the transactions related to furniture and stores in the mining repository and data gets ready to generate rapid reports. With the help of the reports user can get the information on how many people are interested and are willing to buy the furniture.
Impressed with this result, he might plan for another ad for their electronic items, then the administrator configures an electronic item filter. Therefore, we can update or modify these filters as per the usage and requirements. Filters can also be deleted whenever required.

Thus, we have the whole control on the tool, and we can write certain rules according to which the data will be stored.

4.1.1.2. Filtering utilization (Censorship)

Technique of censorship made the tool to overcome challenges of hazardous activities that take place over the internet. The hazardous of "web surfing" include:

❖ Advertisements

The GIFs and JPEGs at the various search engine sites take time to load, fill up screen, and allow third parties to collect information about "web surfing."

❖ Information collection

There is a surprising amount of information that web servers can collect about information requests; the search engine providers are trying to find ways to sell this sort of information.

❖ Sexual, Dissident, Hate-Oriented and Dangerous Material

Mature people with self-control may merely find it "generally offensive," and avoid it when it comes up. Others may be less tolerant, particularly when the "surfer" may be a child.
4.1.2. Rule set

The collection of filters makes up the “rule set” and they have to be defined in the format specified in the tool’s user interface (UI) in proper order.

Basing on Rule set the tool will accept or reject the data.

To ignore unwanted data on the website the tool must be configured with the data categorization filters to allow only required data in to the Web Usage Mining repository. This tool provides an advanced technique to filter the data, since the filters are dynamically configured as per the demands of the business without making the website offline.

4.1.3. Live Data

Live data defined as a data evaluated while the user performing the transaction on the web site.

Live Data has created a new way to address the needs of people and technology to make a real difference in the analyzing world. While the user is performing the transaction on the website “Online Miner” unique feature gathers information from the website and sends the collected information to processes and presents it on repository for rich displays.

Administrator has to define certain rule set to store the data in the required category, if one can’t define what data should be deposited, then the rules cannot be developed to enforce. It provides a distinct feature of storing data into the web usage repository i.e. Data collected from various sources of the page will be stored in a defined order. User’s behavior is stored in the respective categories provided by the administrator.
The rule set should follow a categorization, transforming and filtering of the data, which is an essential component for generating the report and to take the quicker decision on the transaction-taking place on the website.

It's a time saving process, which generates rapid reports, making it easy for analysis and top of all it captures the data live.

4.1.4. Flexibility

This tool is inherently simple, but retains a high degree of flexibility. It neatly surfaces the key concepts most commonly required to control the analysis and delivery of data for categorization.

A shoe company wants to advertise their product on a shopping website, they have to get the data on the frequency of clicks on particular shoe tab in order to advertise their product.

This tool also provides the flexibility to the site administrators to select the required fields to capture the data and store the data according to the user defined setting in the WUM repository.

4.1.5. Performance

This is an Add on tool, which means that the tool can be added on the actual website directly. Since the data is collected according to the rules set by the administrator it minimizes hits to the database by saving only the required data, this in turn increases the performance of the web page.
4.1.6. Security

The tool is integrated with detailed understanding of data networking elements, as well as a detailed understanding of network protocols (the languages used when systems communicate over the network).

Almost all network computing involves one of two types of transmission: file transfer or interactive transmission. A Web server or a file server is a machine that invites other computers to visit and use its resources; this basis itself is insecure. The challenge now is to prevent users from using anything (any resource) on web server and provide only the resources with administrator which have set up for them to access.

A good example is Java. Using Java from the client computer user asks the server for a executable file. This means web server computer is specifically taking direction from client computer. Suppose that the client computer directs server computer to reconfigure its own registry setting or access additional resource from the database; then this activity leads to create a problem. This is an example of a security hole. This could happen involuntarily/voluntarily if we have an incompetent program deployed on the web server, that damages the computer, or it could be a malicious event takes place on the server.

Furthermore, Online Miner has the ability to translate transactions along with a security policy into an effective rule set that can be applied to website.

4.1.7. Individual Focus

If a website contains more number of pages this tool can be configured to collect the data from a specific page or pages, allowing the user to restrict particular page(s) from collecting data, therefore provides individual focus on a website.
4.1.7.1. Scenario

Let us take a scenario that a website contains 100 WebPages and the requirement is to collect the data from one specific page.

The dynamic nature of the tool allows the administrator to configure the rules to meet the above requirements.

The below solid circles indicate the web pages and symbols in the circles represent the controls like text box, chat box, db controls etc., each and every item and page is identified with unique name. Filters are configured to pick the data from individual controls using unique names.

This results in increasing the performance of Online Miner and successfully collecting only the specified/required data.

Fig 4.1: Individual focus on data

This results in increasing the performance of Online Miner and successfully collecting only the specified/required data.
Circles with dotted lines show the specified data or the data required on the web pages. This data is fetched by ignoring the other behavior of the user to the next level of transformation. This architectural design enables the tool with the special feature of lightweight data transactions.

4.1.7.2. Example

On the inventory web page product, product Qty and product code information is available, the tool can be configured to collect only the product code from the web page ignoring rest of the data.

Fig 4.2: Inventory web page with magnifying product code
Instead of configuring each and every control of the page there is also a provision in the tool to ignore only one particular control’s data and accept rest of the data from controls and send to WUM repository.

This is the additional quality of the tool that gives the flexibility to the Admin to configure the filters easily incase of bulk controls on the page.

4.1.8. Transformation through keywords

A data transformation is a mapping procedure that converts data into different representations that may provide more insight for an analyst.

Data transformations are required to convert data into structured forms that permit them to be visualized and analyzed.

Data transformations are used to augment data by deriving additional data. Data transformations convert data into new and meaningful forms. For example, linguistic analysis can be used to assign meaning to the words in a text document.

Data transformations make it possible to create more useful visual representations that support more sophisticated analyses.

Data transformations can be applied iteratively, with each transformation producing a new representation and potentially leading to new insights. Data transformations may be used to find convenient layouts for displays such as by creating a low-dimensional display space from a high dimensional data space.

A major challenge of visual analytics is to find the most useful ways to couple data transformations with interactive visual representations and analytical
reasoning techniques. Data transformation techniques should not introduce biases that would affect analytic conclusions based on the data. At the same time, they should preserve the inherent biases, uncertainties, and other quality attributes of the original data.

4.1.9. Separation of data from presentation

It is widely recognized that an architectural separation between data model and presentation is desirable within all Web applications. For instance, the **Model-View-Controller** paradigm is one widely used approach to achieve this, and the architecture described in this article is compatible with such approaches.

As well as providing a well-defined interface to separate the implementation of data access from that of presentation, the actual tasks of creating such implementations are also separated. Once the initial work of defining the data sources (in terms of dimensions, measures, and so forth) is completed, and then different individuals can do in parallel and the tasks of implementing them and any pages that depend on them.

Tool data model is inherently simple, but retains a high degree of flexibility. It neatly surfaces the key concepts most commonly required to control the analysis and delivery of data for presentation.

Online Miner, in conjunction with a well-implemented data source, provides the page author with little likelihood of inadvertently configuring an incorrect analysis of the data. This is in stark contrast to technologies such as SQL, where a simple typographical error or misunderstanding of the database schema can lead to very significant and sometimes subtle errors.
From the perspective of the data access programmer, the Online Miner provides a lot of flexibility for implementation. It is conceptually simple and requires only the ability to filter and aggregate data to support it. This leaves a considerable design space for the data access programmer to exploit any capabilities of the underlying data source.

4.1.10. Easy Analysis/accurate reports:

Reports are visually appealing and make it easy for users to see comparisons, patterns, and trends in data.

For instance, rather than having to analyze several columns of tables in the database, user can see at a glance whether sales curve is moving upward or falling downward over quarterly periods and how these actual sales compare to the planned sales.

The report is linked to the source data in the WUM repository, which means the report, is updated when data in the WUM live repository is manipulated / added.
4.2 Online Miner High Level Architecture

Fig 4.3: Online Miner Architecture
4.3. Online Miner in action

4.3.1. Collecting Navigation Data

4.3.1.1. User (Visitor)

User activities on the web are the main source of WUM; all the web page presentation layers are designed with HTML, JavaScript, and Flash. WebPages are interpreted and presented in front of the user with the help of Web browsers like Microsoft Internet Explorer, Netscape etc, and Presentation layer of web pages allows the user to interact with WebPages and generates the user behavior.

4.3.1.2. Web Interface

All the web community starts the activities using web interface (formally called as web pages/web portals), this will be a most preliminary level in the WUM. This interface should be more attractive and user friendly. keeping this as a base criteria Online Miner is designed as an add-on tool. Web developers can design the web pages as per the business specifications, later “Online Miner” tool can be integrated with the web pages designed by the designer. Using the web scripting languages “Online Miner” can be incorporated into the web interface.

Web community includes end users and administrator. End users refers to the people consuming the internet services like sending/reading the emails, online shopping etc., Administrator refers to the people managing the web sites on the central servers. General responsibilities of administrator as follows

a) Setting the web site on the server
b) Providing the permission on the web pages
c) Maintaining the backups and restore the website in case of problems

d) Configure the page as per the end user requirements however there will be no impact on the website business policies and guidelines

"Online Miner" facilities the administrator to configure the web settings and define the data filters using web portals (web interface). Furnished web interface enables the administrator to implement setting and configuration very comfortably with minimal technology.

4.3.1.3. User Hits

To navigate the data from one page to another page user hits the hyperlinks, tool captures these hits and stores into the repository according to the user defined rule set.

4.3.1.4. Request/Response

User confirms the data entry by clicking on the submit button, all the data entered by the user collected and sent to the server in the form of request. Server sends back the information to the user according to the request raised by the user. This activity is called as response.

The online feature of the tool starts functioning while the request and response activity takes place on the web page.
4.3.1.5. User identification

In some cases, information collected inside a web server log file cannot be directly used for Web Usage Mining tasks; there are several reasons to prove this fact.

Tool collects the data along with the session information as well as the user information that is tightly integrated with web page.

4.3.2. Parsing Services of Online Miner

Parsing service of Online Miner encapsulates the functionality of controlling seemingly disorganized requested data by applying predefined filters and ignores unwanted data. The organized data leaves the parsing service and it makes its way to the Analysis engine of online Miner and heads to transformation level of online Miner.

4.3.3. Data Regulation (Data mapping dictionary)

4.3.3.1. Keywords

Extraction of relevant key phrases/group of characters from the data limited to the concepts that are pertinent to the topic of the whole document.

4.3.3.2. Mapping

Keywords will be changed according to the user-defined synonyms in the dictionary.
Example:

As shown in the below table 4.1 user (Admin) can define the keywords according to the business requirements / specifications

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Mapped word</th>
</tr>
</thead>
<tbody>
<tr>
<td>Food</td>
<td></td>
</tr>
<tr>
<td>Goat</td>
<td>Meet</td>
</tr>
<tr>
<td>Chicken</td>
<td>Meet</td>
</tr>
<tr>
<td>Beef</td>
<td>Meet</td>
</tr>
<tr>
<td>Pork</td>
<td>Meet</td>
</tr>
<tr>
<td>Medication</td>
<td></td>
</tr>
<tr>
<td>Dettol</td>
<td>Antiseptic</td>
</tr>
<tr>
<td>Savlon</td>
<td>Antiseptic</td>
</tr>
<tr>
<td>Sprit</td>
<td>Antiseptic</td>
</tr>
<tr>
<td>Chatting vocabulary</td>
<td></td>
</tr>
<tr>
<td>Hey</td>
<td>Hi</td>
</tr>
<tr>
<td>heey</td>
<td>Hi</td>
</tr>
<tr>
<td>Gr8</td>
<td>Great</td>
</tr>
</tbody>
</table>

Table 4.1. Table for Keyword specifications

4.3.3.3. Analysis Engine

A bit better-organized analysis is done in this level. The engine picks the data from preceding level and starts looks for the words to match depending up on the keywords are defined in the mapping dictionary. It convert them according to the dictionary.
4.3.4. Generating Data Mining Model

4.3.4.1. Transformation and Categorization

A data transformation is a dynamic procedure that converts data into different representations that may provide more insight for an analyst. Rule set is used to organize data into groups. They divide the data into new and meaningful categories. Transformations make it possible to create more useful visual representations that support more sophisticated analyses.

Data transformations can be applied iteratively, with each transformation producing a new representation and potentially leading to new insights.

Data transformation and categorization techniques in this tool will not introduce issues that would affect the analytic conclusions based on the data. At the same time, they preserve the quality attributes of the original data.

4.3.4.2. Evaluate Data and Archive

Evaluated data is segregated and stored in the Web Usage Repository, ready to generate reports. To maintain and keep up the performance of the tool, it provides an option to the administrator (website technical owner) to decide up on moving the data from live repository to archive (off line repository) for offline analysis and reporting.

Various types of data are present in the WUM repository, and each type must be represented in ways that facilitate computational transformations to yield analytical insight. Visualization that combines multiple types of data is also needed to support comprehensive analytic reasoning in certain situations.
The data representation must faithfully preserve the inherent structure of the data as well as the data’s real world context and meaning. It is important for the data representation to portray the quality of the data as collected. If information is missing, purposefully hidden, or misleading, then technical analysis must be done to choose the proper representation for the data.

**4.3.4.3. Mining repository algorithm**

**Quick mining repository algorithm**

**Step 1**
Collection of data from live source

**Step 2**
Key phrases transformation/ character transformation

**Step 3**
Identifying the data into category (Cluster Model)

**Step 4**
Stores the data in the tables with references of category, it will be moved into the archive locations basing on administration action.

**Step 4**
Report generation
4.3.5. Final Report

Reports are visually appealing and make it easy for users to see comparisons, patterns, and trends in data. For instance, rather than having to analyze several columns of tables in the database, user can see at a glance whether sales curve is moving upward or falling downward over quarterly periods and how these actual sales compare to the planned sales.

The report is linked to the source data in the WUM repository, which means the report, is updated when data in the WUM live repository is manipulated / added. Sample report can be seen in Sample Reporting section.
4.4. Technical summary

4.4.1. C# features

C# (pronounced "C-sharp") is an object-oriented programming language from Microsoft that aims to combine the computing power of C++ with the programming ease of Visual Basic. C# is based on C++ and contains features similar to those of Java.

C# is designed to work with Microsoft's .NET platform. Microsoft's aim is to facilitate the exchange of information and services over the Web, and to enable developers to build highly portable applications. C# simplifies programming through its use of Extensible Markup Language (XML) and Simple Object Access Protocol (SOAP) which allow access to a programming object or method without requiring the programmer to write additional code for each step. Because programmers can build on existing code, rather than repeatedly duplicating it, C# is expected to make it faster and less expensive to get new products and services to market.

- The first component oriented language in the C/C++ family
- Everything really is an object
- Next generation robust and durable software
- Preservation of investment
- Component concepts are first class:
  - Properties, methods, events
  - Design time and run time attributes Design-run-
  - Integrated documentation using XML
- Enables one stop programming one-
  - No header files, IDL, etc.
  - Can be embedded in web pages

- Everything really is an object
  - C++, Java: Primitive types are “magic” and do not interoperate with objects
  - Smalltalk, Lisp: Primitive types are objects, but at great performance cost
  - C# unifies with no performance cost
  - C# maintains deep simplicity throughout system
  - C# provides Improved extensibility and reusability
  - C# with New primitive types: Decimal, SQL… Collections, etc., work for all types
  - Robust and durable software

- Garbage collection
  - No memory leaks and stray pointers
- Exceptions
  - Error handling is not an afterthought

- Type safety Type-
  - No uninitialized variables, unsafe casts
- Versioning
  - Pervasive versioning considerations in all aspects of language design
4.4.2 What is .NET?

When .NET was announced in late 1999, Microsoft positioned the technology as a platform for building and consuming Extensible Markup Language (XML) Web services. XML Web services allow any type of application, be it a Windows- or browser-based application running on any type of computer system, to consume data from any type of server over the Internet. The reason this idea is so great is the way in which the XML messages are transferred: over established standard protocols that exist today. Using protocols such as SOAP, HTTP, and SMTP, XML Web services make it possible to expose data over the wire with little or no modifications to your existing code.

Below picture presents a high-level overview of the .NET Framework and how XML Web services are positioned.

Stateless XML Web services model.

Fig 4.5 Stateless XML Web Services Model
Since the initial announcement of the .NET Framework, it's taken on many new and different meanings to different people. To a developer, .NET means a great environment for creating robust distributed applications. To an IT manager, .NET means simpler deployment of applications to end users, tighter security, and simpler management. To a CTO or CIO, .NET means happier developers using state-of-the-art development technologies and a smaller bottom line. To understand why all these statements are true, you need to get a grip on what the .NET Framework consists of, and how it's truly a revolutionary step forward for application architecture, development, and deployment.

### 4.4.2.1 Asp.net dealing with the Web Request

The best way to understand the internals of the request processing architecture in ASP.NET is to trace the path of a request from its initial entry into a server all the way to its ultimate dispatching to the ProcessRequest method of your Page class. We begin with a high-level overview of how requests are routed on a machine processing requests with ASP.NET, and then take a more detailed look at the internals of the pipeline.

### 4.4.2.2 View of Request Processing

Typically, ASP.NET requests initiate from IIS. When an HTTP request comes in from a client, typically on port 80, the IIS process (inetinfo.exe) receives the request and attempts to locate an extension mapping for the URL requested.

If the request is for an ASP.NET page (it ends with .aspx), IIS loads the aspnet_isapi.dll ISAPI extension DLL and passes the request on to it. Once the aspnet_isapi.dll receives the request, it attempts to locate the ASP.NET worker process, housed in aspnet_wp.exe. If it is not running, it is started, and several
named pipe connections are established between the ISAPI DLL and the worker process.

Once these connections are established, the request is sent across a named pipe to the ASP.NET worker process for handling. Inside the worker process, ASP.NET routes the request to a designated AppDomain and dispatches it to the HTTP pipeline in that AppDomain. The end result of the request passing through the pipeline is the compilation (first time only) and creation of a class that implements the IHttpHandler interface, typically your Page-derived class. This handler acts as the endpoint of the request, populates the response buffer, and delivers the response back through the same channels it came from until IIS ends up sending back the response buffer. This depicts this high-level view of request processing in ASP.NET. It is possible to host ASP.NET using another mechanism for dispatching and processing requests. Microsoft has produced a sample called Cassini, which demonstrates hosting ASP.NET in a custom Web server application. For more details, see http://www.asp.net.

4.4.23. High-Level View of Request Processing in ASP.NET

![High-Level View of Request Processing in ASP.NET](image)

Fig 4.6 High Level View of Request Processing

71
4.4.2.4. Inside the Pipeline

Once the request makes it into the worker process, it goes through a series of steps and classes before it arrives at the ultimate handler. To begin with, each ASP.NET application is housed within an AppDomain in its worker process. AppDomains are a CLR construct that provides processlike memory and security isolation, without the overhead of actually creating separate processes. ASP.NET uses AppDomains to separate applications from each other so that if one application has a problem, it can safely be removed without affecting the remaining applications. Depicts the various pipeline classes that live within an application's AppDomain and shows how they interact to service a request.

4.4.2.5. Classes in the HTTP Pipeline

Fig 4.7. Classes in the HTTP Pipeline
The first thing that happens when a request is dispatched to an application is that an instance of the HttpWorkerRequest class is created (1), which contains all the information about the current request, including the requested URL, the headers, and so on.

Once the HttpWorkerRequest class is created, it is passed into the static ProcessRequest method of the HttpRuntime class, which is executed in the AppDomain of the application, initiating the processing of the request (2).

The first thing the HttpRuntime class does is to create a new instance of the HttpContext class, initialized with the HttpWorkerRequest class (3).

The HttpContext class is the "glue" of the pipeline, since it holds all the classes together by keeping all the relevant information about the current request in one location.

When the HttpContext class is first created, it allocates new instances of the HttpRequest and HttpResponse classes and stores them as fields. It also provides property accessors to the application and session state bags.

Once the HttpContext class is created, the HttpRuntime class requests an instance of the HttpApplication-derived class for this application by calling the static GetApplicationInstance method of the HttpApplicationFactory class (4).

GetApplicationInstance either creates a new instance of the HttpApplication (or a derivative) class or pulls one from a pool of application objects if one is available (4).
Once the HttpApplication class is created or retrieved, it is initialized, and during its initialization it allocates any modules that are defined for this application (6). Modules are classes that implement the IHttpModule interface and serve to pre- and postprocess requests.

Once the modules have been created, the HttpRuntime class asks its newly retrieved HttpApplication class to service the current request by calling its BeginProcessRequest method (7), defined by the IHttpAsyncHandler interface implemented by the application class.

The HttpApplication class then takes over the request processing and locates the appropriate handler factory for the current request, based on the URL path.

For example, if the request is for an .aspx page, it uses the PageHandlerFactory class. Once it locates the appropriate factory, it invokes the GetHandler method on the IHttpHandlerFactory interface to retrieve a fresh copy of the appropriate handler class. Handler classes serve as the endpoint for requests and very often are simply the Page-derived class that is created from an .aspx file.

In general, handlers are classes that implement the IHttpHandler interface and populate the response buffer when asked to process a request. Once the handler is created, its ProcessRequest method is called (8), passing in the current HttpContext class so that it has access to the Request, the Response, and all the other request-specific pieces of information necessary. Once the ProcessRequest method returns, the request is complete.
4.4.2.6. Controls used for developing administration user interface

- HTML server controls HTML elements exposed to the server so you can program them. HTML server controls expose an object model that maps very closely to the HTML elements that they render.

- Web server controls Controls with more built-in features than HTML server controls. Web server controls include not only form controls such as buttons and text boxes, but also special-purpose controls such as a calendar, menus, and a tree view control. Web server controls are more abstract than HTML server controls in that their object model does not necessarily reflect HTML syntax.

- Validation controls with incorporated logic enable you to control what the users enter for input in the TextBoxes. Validation controls enable you to check for a required field, to test against a specific value or pattern of characters, to verify that a value lies within a range, and so on.

4.4.2.7. Communication with the database using ADO.Net

ADO.NET provides consistent access to data sources such as Microsoft SQL Server, as well as data sources exposed through OLE DB and XML. Data-sharing consumer applications can use ADO.NET to connect to these data sources and retrieve, manipulate, and update data.

ADO.NET cleanly factors data access from data manipulation into discrete components that can be used separately or in tandem. ADO.NET includes .NET Framework data providers for connecting to a database, executing commands, and
retrieving results. Those results are either processed directly, or placed in an ADO.NET `DataSet` object in order to be exposed to the user in an ad-hoc manner, combined with data from multiple sources, or remoted between tiers. The ADO.NET `DataSet` object can also be used independently of a .NET Framework data provider to manage data local to the application or sourced from XML.

The ADO.NET classes are found in System.Data.dll, and are integrated with the XML classes found in System.Xml.dll. When compiling code that uses the `System.Data` namespace, reference both System.Data.dll and System.Xml.dll. For an example of compiling an ADO.NET application using a command line compiler

ADO.NET provides functionality to developers writing managed code similar to the functionality provided to native COM developers by ADO. For a discussion of the differences between ADO and ADO.NET

4.4.2.8. User ID and security related transactions utilized in session objects

ASP.NET session state enables you to store and retrieve values for a user as the user navigates ASP.NET pages in a Web application. HTTP is a stateless protocol. This means that a Web server treats each HTTP request for a page as an independent request. The server retains no knowledge of variable values that were used during previous requests. ASP.NET session state identifies requests from the same browser during a limited time window as a session, and provides a way to persist variable values for the duration of that session. By default, ASP.NET session state is enabled for all ASP.NET applications.

Alternatives to session state include the following:

- Application state, which stores variables that can be accessed by all users of an ASP.NET application.
- Profile properties, which persists user values in a data store without expiring them.
- ASP.NET caching, which stores values in memory that is available to all ASP.NET applications.
- View state, which persists values in a page.
- Cookies.
- The query string and fields on an HTML form that are available from an HTTP request.

Above information is taken from the base source of Microsoft.

4.4.3. Glance on HTML

One of the original design goals of HTML was to be \textit{device independent}. That is, it was to be used on a variety of computer systems without change. Accordingly, it was designed to be a language to describe document structure, rather than document presentation. So the basic HTML elements specify such things as headings, titles, and paragraphs - but not margins and fonts. It was left up to the browser on any specific system to take care of rendering the document in whatever way the browser author thought most suitable.

This was appropriate for the original target audience - scientific researchers who were far more interested in what was in a document, than how it appeared. But the WWW concept rapidly became very popular and other communities - especially the 'commercial' people - became interested in it. And in the commercial world, image is very important. If your documents are going to be seen by thousands or millions of people, they had better look good. But the original HTML offered very
little support for layout and presentation, so the demand grew for extensions. Various browser manufacturers have introduced new HTML elements oriented towards presentation issues - notably Netscape - and some of these have been adopted into the HTML standards proposals.

Putting presentation instructions directly into documents goes against the original device-independence goal of HTML, and makes it hard to change the presentation later. The correct solution, called style sheets, has been known from before the beginning of the WWW project. Soon after its formation in mid '94, the W3C published the first draft of a concrete proposal, and early implementations were pioneered by the Arena and emacs-w3 browsers in the beginning of '94. IIRC, Navipress, Panorama, and maybe a few other commercial browser/editor tools also had some style sheet support at around the same time. The first 'mass-market' browser to adopt the stylesheet approach was MSIE, about a year later. While you can most certainly stretch the features of HTML to provide useful layout and presentation, the trend is going to be towards moving that aspect into style sheets.
### 4.4.4. Table and Fields Description

<table>
<thead>
<tr>
<th>Column</th>
<th>Purpose</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>RequestString</td>
<td></td>
<td>This store the request string taken from the user request (as-is data will be stored in this record)</td>
</tr>
<tr>
<td>Date</td>
<td></td>
<td>It stores the date and time of the request received from the user</td>
</tr>
</tbody>
</table>

**Table 4.2. Table for MiningSource**

<table>
<thead>
<tr>
<th>Column</th>
<th>Purpose</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeyWord</td>
<td>PrimaryKey</td>
<td>Stores the key word matches a particular category</td>
</tr>
<tr>
<td>CategoryRef</td>
<td>To store the category reference ID</td>
<td>This key a reference key to store the category, this identifies keyword to be mapped to a particular category</td>
</tr>
<tr>
<td>IsAccept</td>
<td>Boolean</td>
<td>Default value of this field is true if the value is set to false then keyword will be removed while moving to the source table</td>
</tr>
</tbody>
</table>

**Table 4.3. Table for KeyWord**
### Table 4.4. Table for KeywordTransformation

<table>
<thead>
<tr>
<th>Column</th>
<th>Purpose</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>TmsWord</td>
<td>PrimaryKey</td>
<td>Stores the key word matches a particular category</td>
</tr>
<tr>
<td>KeyWord</td>
<td>To store the Keyword reference ID</td>
<td>This key, a reference key to convert the similar words to one word t. This identifies the shortcut words that are to be mapped to a particular keyword</td>
</tr>
</tbody>
</table>

### Table 4.5. Table for Filter Accept

<table>
<thead>
<tr>
<th>Column</th>
<th>Purpose</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>PageName</td>
<td>Program name of the page</td>
<td>(if the lookup page name not found by default all the controls data to be accepted)</td>
</tr>
<tr>
<td>ControlName</td>
<td>controlID</td>
<td>Program name of the control</td>
</tr>
<tr>
<td>StartWith</td>
<td>Search with wild cards characters</td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If all controls data has to accept starting with “txtStr” then this field has to fill with txtStr</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this column is empty then it will move to next column “Contains”</td>
</tr>
<tr>
<td>Contains</td>
<td>Search with wild cards characters, If this column is empty then it will move to next column “EndWith”</td>
<td></td>
</tr>
<tr>
<td>EndWith</td>
<td>Search with wild cards characters</td>
<td></td>
</tr>
</tbody>
</table>

80
<table>
<thead>
<tr>
<th><strong>Column</strong></th>
<th><strong>Purpose</strong></th>
<th><strong>Remarks</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>PageName</td>
<td>PrimaryKey</td>
<td>Program name of the page</td>
</tr>
<tr>
<td>ControlName</td>
<td>controlID</td>
<td>Program name of the control</td>
</tr>
<tr>
<td>StartWith</td>
<td></td>
<td>Search with wild cards characters</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If all controls data has to reject starting with “txtStr” then this field</td>
</tr>
<tr>
<td></td>
<td></td>
<td>has to fill with txtStr</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this column is empty then it will move to next column “Contains”</td>
</tr>
<tr>
<td>Contains</td>
<td></td>
<td>Search with wild cards characters</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this column is empty then it will move to next column “EndWith”</td>
</tr>
<tr>
<td>EndWith</td>
<td></td>
<td>Search with wild cards characters</td>
</tr>
</tbody>
</table>

**Table 4.6. Table for Filter Reject**
<table>
<thead>
<tr>
<th>Column</th>
<th>Purpose</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>CategoryID</td>
<td>Primary Key</td>
<td></td>
</tr>
<tr>
<td>Category</td>
<td></td>
<td>Category Label will be stored in the column</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Example:</strong> Happy new year wishes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If any keyword which contains Happy new year or new year greetings or best wishes for new year etc will come into this category</td>
</tr>
</tbody>
</table>

Table 4.7. Table for Category

<table>
<thead>
<tr>
<th>Column</th>
<th>Purpose</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>CategoryID</td>
<td>Category Reference</td>
<td></td>
</tr>
<tr>
<td>KeyWordCount</td>
<td></td>
<td>It store the count of findings of the key words related to the particular category</td>
</tr>
<tr>
<td>Date</td>
<td>Date&amp;Time</td>
<td>It stores the date and time. Copy the date from the source table</td>
</tr>
<tr>
<td>KeyWord</td>
<td></td>
<td>Keyword details</td>
</tr>
<tr>
<td>UserID</td>
<td></td>
<td>User details</td>
</tr>
</tbody>
</table>

Table 4.8. Table for Transaction
<table>
<thead>
<tr>
<th>Column</th>
<th>Purpose</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>HyperLinkDesc</td>
<td>Store the link information (optional)</td>
<td></td>
</tr>
<tr>
<td>HyperLinkID</td>
<td>It store the linkID (Which is set in the program )</td>
<td></td>
</tr>
<tr>
<td>Date</td>
<td>It stores the date and time of the request received from the user</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.9. Table for HitSource

<table>
<thead>
<tr>
<th>Column</th>
<th>Purpose</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>CategoryID</td>
<td>Primary Key</td>
<td>Store category ID</td>
</tr>
<tr>
<td>Category Name</td>
<td></td>
<td>Category Label will be stored in the column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hyperlinked: hprAdd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Category: Advertise</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If user click hprAdd, then it will be moved in to Advertise category.</td>
</tr>
</tbody>
</table>

Table 4.10. Table for Hitcategory
### Table 4.11. Table for HyperLink

<table>
<thead>
<tr>
<th>Column</th>
<th>Purpose</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>HyperLinkID</td>
<td>Unique</td>
<td>Store the link information</td>
</tr>
<tr>
<td>HyperLinkDesc</td>
<td></td>
<td>Store the description of the hit</td>
</tr>
</tbody>
</table>

### Table 4.12. Table for HitTransaction

<table>
<thead>
<tr>
<th>Column</th>
<th>Purpose</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>CategoryID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date</td>
<td>Date&amp;Time</td>
<td>It stores the date and time. Copy the date from the source table</td>
</tr>
<tr>
<td>HyperLinkDesc</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UserID</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 4.13. Table Categorization

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>MiningSource</td>
</tr>
<tr>
<td>Evaluate and Archive</td>
<td>Transaction</td>
</tr>
<tr>
<td>Transformation</td>
<td>FilterReject</td>
</tr>
<tr>
<td></td>
<td>FilterAccept</td>
</tr>
<tr>
<td>Data mapping dictionary</td>
<td>KeyWord</td>
</tr>
<tr>
<td></td>
<td>Category</td>
</tr>
</tbody>
</table>
Fig 4.8. Data Base Layout Category
### 4.4.5. Base Class details

#### 4.4.5.1. Base class used in building the tool is described as follows

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBClass</td>
<td>To handle all the database connections and database activities</td>
</tr>
<tr>
<td>MiscTool</td>
<td>Generic tools which helps the application to avoid the redundancy coding</td>
</tr>
<tr>
<td>CopyRequest</td>
<td>This is the main class for building the mining source data.</td>
</tr>
<tr>
<td>TransformView</td>
<td>This class will be used in various purposes to view the report and other activities</td>
</tr>
<tr>
<td>ViewClass</td>
<td>This will present the data in the report view</td>
</tr>
</tbody>
</table>

*Table 4.14. Table for Base Class details*
4.4.5.2. CopyRequest Overview:

- **ReadQueryString**

  Verifies query string object if query string object contains data then call the formatQuerySting to format the data according to the configuration settings, formatted data will be stored in a array variable one line of array will be treated as one record and it will be inserted in to the MiningSource table using database insert method this method will be called at the beginning of the page load event.

- **Configuration settings**

  A configuration setting contains information regarding the words to be removed, fields to be accepted, fields to be ignored and to split the words of the string (Split words settings specified in the web.config).

- **FormatQueryString**

  Go to the FilterReject
  
  Search for the current page in the PageName Column
  
  Ignore the controls according settings available in the FilterReject table

  **Note:**
  
  Rule to apply the reject filter
  
  If the “ControlName” field contains the data then rest of the two fields data will not be considered for filters. Out of the four fields (ControlName, StartWith, Contains, EndWith) only one field data will be considered per record according the order. ControlName will be the starting.
Example

There are five controls on the page

Page name: pgDemo

Controls:

txtBmessage
txtMvalu
RtfTxt
ChkValue
ChkGtr
MNKTxt

From the above controls if there is a requirement to ignore the txtBmessage and controls ending with Txt, then there should be two records. First record “ControlName” column, data should be “txtBmessage” and second record “EndWith” column with “Txt”. Rest of the columns should be blank. Same rule to be applied for FilterAccept. FilterAccept will be implemented after implementing FilterReject.