OLAP/EIS Tops Off the Data Warehouse

Presented By:
Henry M. Stewart
Director of Planning and Data Administration
University of Rochester
(716)275-8039
henrys@acs.rochester.edu

ABSTRACT
On-Line Analytical Processing (OLAP) has emerged as a "break through" technology that can provide the foundation for fast, easy, affordable EIS solutions. OLAP technologies, when integrated with a data warehouse, can aggregate vast amounts of legacy data in both graphic and tabular formats. OLAP technologies feature very fast response, drag-and-drop navigation, point-and-click drill-down/drill-up among levels of detail, and value-based color highlighting for "exception reporting". Using OLAP/EIS solutions, senior managers are able to interactively "browse" hundreds of displays that present a visualization of their institution's business process.

This presentation begins with a quick overview of the University of Rochester's approach to data administration including the development of a multi-gigabyte data warehouse. Distinctions between data querying, reporting, and analysis will be presented. OLAP will be featured as the newest form of these information services. Detailed information and a demonstration will highlight this session.
Information Technology Architecture: An Overview

In the Spring of 1991 the University of Rochester's Administrative Computing Services (ACS) launched a major initiative that responded to the growing need for access to information. ACS recognized that in order to both maintain a stable, responsive, secure environment for its complex enterprise-wide applications (financial, human resource, student, development) and to begin providing users with fast, easy access to data of university-wide interest that it would have to break away from the concept that all computing and data would be maintained on one large mainframe computer system.

To maintain the production environment while moving ahead with new information retrieval services and technologies, a three-tiered Information Technology Architecture was adopted.

IT Architecture: Logical View

IT Architecture: Physical View
Along with a new architecture, ACS adopted a phased approach to the task of providing fast, easy access to university data.

**Phased Approach To Data Warehouse and EIS**

Phase 1: **DO SOMETHING!** Develop a warehouse using character-based, terminal-host technologies that could use the Rolm Data Lines that connect most offices.

Phase 2: After ethernet connections were deployed to most desktops, convert to relational and client-server technologies for query/report services.

Phase 3: Develop some type of support for senior management.

**Warehouse Application Created**

According to the plan, Phase 1 was completed using available technologies and a warehouse was developed with data from the University’s centralized financial accounting system. Tools used were Focus report-writer and the Focus data base. The warehouse was designed to hold data for the University’s seven schools, hospital, and central administration. The Chart of Accounts contains 8,000 unique 6-digit accounts and 100,000 unique 10-digit account-subcode combinations. Each month a snapshot of the financial system was captured and loaded into the warehouse. Approximately 200,000 records are added each month, six million records per year. The current size of our financial data base in the warehouse is approximately 5 gigabytes.

These concepts are illustrated on the next page.
Conversion to Client-Server

Although Focus was fast and worked well for some people, it was too technical for most office staff to learn. It became an extremely valuable tool for information analysts in central offices such as budget, audit, and financial planning but its learning curve was too difficult for occasional users. Its use as a tool for senior management was not even considered. We accepted this condition as a consequence of the technology and waited patiently for client-server technology to mature.

Eventually our campus network expanded and many offices received ethernet connections. Over time we selected Sybase as the relational data base for our warehouse and began converting data from Focus to that platform. As soon as data became available in Sybase we began testing the new toys (i.e. tools) that were available for query and reporting.

Two items became quickly obvious. First, the drag-and-drop, point-and-click, Graphic User Interface (GUI) provided by client-server tools was very attractive, especially when compared to the character-based technologies that were previously installed. Second, performance was slower than with Focus, much slower for some forms of queries.

We spent many weeks experimenting with various tools and forms of data base design ranging from fully normalized to fully denormalized designs. Eventually we settled on a design that gave us good response for most department-level queries. An illustration of a typical query is presented on the next page.
Typical Query Against 5 Gigabyte Data Warehouse (4 second response)

OLAP/EIS Tops Off The Data Warehouse
Presented By: Henry Stewart, University of Rochester CAUSE95

Typical Query Against 5 Gigabyte Data Warehouse (4 second response)
Performance: Query/Report vs. Analysis

Response to data retrieval requests varies widely depending on at least two major factors; data base design and the type of request submitted. In terms of the data base, we found that a denormalized design with many simple and compound indices works best. One year ago Ralph Kimball, former President of Red Brick Systems, introduced the notion of a star-schema. Since that time, a number of other articles have appeared praising this design for data warehouse applications. Although we have not tested this approach, it certainly looks very interesting.

The type of retrieval requested can also make a very big difference in performance. The most common form of a request is one that returns all detail records matching a particular set of conditions. For example, request all subcode records that are year=1996, month=07, account=211706. We refer to these requests as queries or reports. Query/report requests that retrieve data for a specific period of time can run within a few seconds.

Most of the tools that are advertised today are query/report tools. They work best when the user ‘filters’ (i.e. applies specific conditions) the query which limits the number of records that are retrieved. A small result set is returned to the client workstation where the records are formatted into reports.

In order to expect a timely response, users must attach filters to restrict the size of the result set returned from the warehouse. In fact, we encourage new users to ‘over filter’ their requests to create a small result set until all formatting changes are applied. Once a report is formatted correctly, then remove the excessive filtering to retrieve the desired result.

With the development of data warehouses, another form of request has become very popular, the request for analysis. Analysis requests summarize and aggregate vast amounts of data. Results from analysis requests often appear formatted in a cross-tabular or spreadsheet format. The result set from an analysis query could be very large.

Until just recently, if people wanted to produce analysis using client-server technologies, they would often attempt to use query/report tools to perform the task. The very first time that would be attempted, the performance problem would become apparent. Most client-server query/report products don’t respond well to large result sets begin returned to the client, especially if they have to be further aggregated on the client.

Queries that request data from many months or years can take a long time (i.e. hours) to complete. Often Data Base Administrators (DBAs) will be asked to pre-aggregate data into summary tables in order to reduce this retrieval time. Although DBAs need to be able to ‘tune’ a warehouse to respond to common requests, asking them to out-guess management by creating all of the summary tables they might want is probably not a good use of their time.

An example might help emphasize this important point. The query illustrated on the previous page was very responsive only because we asked for a specific account, month, and year. If we remove the filter for month (i.e. ask for account and subcode data from all months) the response may be very different. The data base optimizer may determine that the number of records that will be returned will be large and so it may change its retrieval mechanism and response time could increase to 3-5 minutes. Summary reports for senior management that pull data for multiple departments or an entire school over a two year period can take 20-40 minutes. University-wide reports that produce multi-year trends can take 1-2 hours to complete!
This is a disappointing situation. We want to provide management with the ability to leverage the full potential of historical data but that will not happen with standard query/report tools. In fact, we need to limit the number of analysis queries that retrieve historical data because those queries pull down warehouse performance for the entire community. Moving management queries to off hours might work well but our attempts at convincing senior management to work nights and weekends haven’t been successful. So ... until we find a way to access vast amounts of historical data in the same fast, easy manner that we query/report data, management will not be able to maximize the value of this valuable resource.

This is where the University of Rochester found itself exactly one year ago. At that time, the only way we thought we would be able to leverage the great value of our historical data would be to extract subsets for specific departments or vice presidents. The plan was to pre-process this data, compress it, aggregate it and then place it on their department server.

Thank goodness we didn’t get into that game. OLAP emerged just in time.

**OLAP To The Rescue**

In November of 1994, while testing client-server query/report tools, ACS began to hear about a new technology called On-Line Analytical Processing (OLAP). Observing with a casual interest, we were convinced to investigate the new technology after reading an article by Richard Finkelstein titled “Understanding the Need for On-Line Analytical Servers” (see suggested readings at the end of this paper). Several months later, after pursuing several vendors and products, we selected PowerPlay from Cognos for the development of a prototype.

During the Spring of 1995 we began demonstrating PowerPlay to senior management and other IS staff members. Their reactions were all very similar - - spontaneous, short bursts of uncontrolled laughter. They couldn’t believe what they were seeing! PowerPlay’s OLAP technology was giving us instant access to aggregates from millions of records from our warehouse. The product provided one second response to hundreds of graphic and tabular displays, with drill-down/drill-up between levels of detail and a mouse oriented drag-and-drop, point-and-click interface that was so easy to use, even senior managers could use the product!

Many articles have since been published that explain the features of OLAP, how they work, their costs, and the vendors that have products on the market. A list of several articles that we found to be helpful is presented at the end of this paper.

**How does it work?**

OLAP technology consists of two major components, the server and the client. Typically the server is a multi-user, LAN-based data base that is loaded either from your legacy systems or from your data warehouse. You don’t need a data warehouse in order to implement OLAP but if you have historical data, OLAP’s visualization capabilities will reveal patterns of your business process that are hidden in the data.

**The Server**

<table>
<thead>
<tr>
<th>Types of Queries/Reports</th>
<th>Speed of Queries/Reports</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Simple Lists</td>
<td>• One Month, One Dept/Acct (fast)</td>
</tr>
<tr>
<td>• Formatted Reports With Sort</td>
<td>• One Month, All Accts (slower)</td>
</tr>
<tr>
<td>• Groups With Subtotals</td>
<td>• All Months, One Act (even slower)</td>
</tr>
<tr>
<td>• Summary Reports, Crosstabs</td>
<td>• All Months, All Acts (<em>very</em> slow)</td>
</tr>
<tr>
<td>• Executive Summaries</td>
<td>• All Months, All Depts, All Acts (<strong>go to lunch</strong>)</td>
</tr>
</tbody>
</table>

- **Simple Lists**
- **Formatted Reports With Sort**
- **Groups With Subtotals**
- **Summary Reports, Crosstabs**
- **Executive Summaries**

- **One Month, One Dept/Acct (fast)**
- **One Month, All Accts (slower)**
- **All Months, One Act (even slower)**
- **All Months, All Acts (**very** slow)**
- **All Months, All Depts, All Acts (**go to lunch**)**
Think of OLAP data bases as multi-dimensional arrays or cubes of data, actually cubes of cubes, capable of holding hundreds of thousands of rows and columns of both text and numbers. The current terminology for these data base servers is Multi-Dimensional Databases (MDDs). The MDDs are loaded from your data source (legacy or warehouse) according to an aggregation model that you define. Fortunately, defining the model and loading the data base can be very easy. For some OLAP products absolutely no programming is required to build the model or to load the data.

The Client
The client component for several OLAP products presents a spreadsheet-type interface with very special features. Features available in some products include the ability to instantly change the data component of either the x, y, or z dimension of your spreadsheet using drag-and-drop. You can change your display from tabular to any one of various charts including pie, bar, stacked bar, clustered bar, line, or multi-line - all with one second response using drag-and-drop. Exception highlighting is another very nice feature. This allows your display to dynamically change font, point-size, and color of rows or columns based upon the value of a component of the display. You can also hide rows or columns based on dynamic values.

Instant drill-down/drill-up is a particularly valuable feature. For example, placing your mouse over a school name and double clicking can invoke a drill-down to department-level numbers. Double clicking on a department can take you down to account-level data. At any point you can then jump from one graphic or tabular display to the next, including multi-year displays, each with a one second response time.

Rapid Development
As stated above some OLAP products require absolutely no ‘programming’ in order to define an MDD model or to load the data. At the University of Rochester, the average time required to create a model is 20 minutes. The average time to load an MDD is another 20-40 minutes. Therefore we tell our customers “Once you give us a data file, we will have you paging through graphics within one hour.” With such rapid development capabilities it is possible to think of EIS solutions that address a short-term business opportunity. In fact, disposable solutions can be considered. Develop a system, use it for 2-4 weeks and then discard it.

Interactive Scenario: Create An OLAP Application
At this point in the presentation, based upon our knowledge of typical financial systems, we will simulate the construction of an OLAP application. In the process of constructing this prototype, several fundamental OLAP concepts will become clear, including the concepts of data dimensions, Multi-dimensional Data Base (MDD), hierarchical data models and steps involved in moving data from a warehouse into an OLAP application.

Our objective is to create an OLAP application that will allow management to view a wide variety of spreadsheets and graphic displays of financial data. Management wants to see financial aggregates by year, month, school, department, account, subcode, ledger, and various other attributes. They need to see dollars and percents sliced-and-diced by those dimensions. We need to extract data from our Financial Accounting System (MVS mainframe) and load it into an OLAP data base located on our departmental server. We need to:

Identify and Extract Data To Be Used
Identify what views (dimensions) will be most important.
( create diagrams of typical spreadsheets, graphs )
Identify what data must be extracted to produce those views (dimensions).
Write a 4GL program to extract the data into a large flat, delimited ASCII text file.

Build an OLAP model
Identify the data dimensions that will be used.
Identify what data items from the extract file will be loaded into what dimensions.
Identify what data items will be measured (added, totaled, summed).

Load The Application
Now we are ready for the final step, load the OLAP data base with data from the extracted file according to the model.

Creating An OLAP Application

Data Warehouse
Financial Data

Flat File Extract of Data

- div,dpt,act,sub,--,--,$,$
- div,dpt,act,sub,--,--,$,$
- div,dpt,act,sub,--,--,$,$
- div,dpt,act,sub,--,--,$,$
- div,dpt,act,sub,--,--,$,$

... and so on ...

OLAP Data Model (Dimensions)

- School: Sch, Dpt, Account
- Time: Year, Qtr, Month
- Subcodes: Rev/Exp, Group
- Ledgers
- Other Attributes?
  - fund groups
  - responsible person

Departmental Server
Multidimensional Data Base (MDD)

(OLAP tool loads flat data file into MDD according to rules defined in data)

Power-Play
**PowerPlay: An OLAP Tool In Action**

Here is an example of an OLAP tool in action, PowerPlay from Cognos. During the live presentation of this paper, many screens will be displayed. Only a few are included in the hard-copy version of this report.

Here is a very brief example of a realistic situation. A senior officer needs to do a quick evaluation of Travel and Conference expenditures. The manager launches PowerPoint, selects an MDD, and begins browsing aggregates.

“Let me look at the University’s Travel and Conference expenditures.”

**Travel and Expense By College (one second)**

<table>
<thead>
<tr>
<th>College</th>
<th>Travel and Conference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administration</td>
<td>$1,611,564</td>
</tr>
<tr>
<td>International Studies</td>
<td>$793,666</td>
</tr>
<tr>
<td>Humanities Center</td>
<td>$1,707,782</td>
</tr>
<tr>
<td>School of Engineering</td>
<td>$893,841</td>
</tr>
<tr>
<td>Medical School</td>
<td>$683,555</td>
</tr>
<tr>
<td>School of</td>
<td>$52,410</td>
</tr>
<tr>
<td>School of Education</td>
<td>$318,878</td>
</tr>
<tr>
<td>Nursing School</td>
<td>$3,078,426</td>
</tr>
<tr>
<td>Graduate Studies</td>
<td>$1,367,168</td>
</tr>
<tr>
<td>Nursing</td>
<td>$159,660</td>
</tr>
</tbody>
</table>
“Let’s take a closer look at the Humanities Center.”

<table>
<thead>
<tr>
<th>Department</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asian Art</td>
<td>na</td>
</tr>
<tr>
<td>Hindu Religion</td>
<td>$11,504</td>
</tr>
<tr>
<td>Buddhist Studies</td>
<td>$8,852</td>
</tr>
<tr>
<td>Taverns of Bombay</td>
<td>$807,399</td>
</tr>
<tr>
<td>New Delhi Temples</td>
<td>$123,624</td>
</tr>
<tr>
<td>Old Delhi</td>
<td>$93,451</td>
</tr>
<tr>
<td>Floral Patterns</td>
<td>$53,382</td>
</tr>
<tr>
<td>Woodstock School</td>
<td>$5,313</td>
</tr>
<tr>
<td>Herbal Medicine</td>
<td>$109,559</td>
</tr>
</tbody>
</table>
“I wonder which departments are spending the most.”

<table>
<thead>
<tr>
<th>Department</th>
<th>Rank (94)</th>
<th>FY94</th>
</tr>
</thead>
<tbody>
<tr>
<td>Christian Influence</td>
<td>1</td>
<td>$507,309</td>
</tr>
<tr>
<td>World Religions</td>
<td>2</td>
<td>$316,392</td>
</tr>
<tr>
<td>Asian Temples</td>
<td>3</td>
<td>$163,742</td>
</tr>
<tr>
<td>New Delhi Temples</td>
<td>4</td>
<td>$123,624</td>
</tr>
<tr>
<td>Bombay Taverns</td>
<td>5</td>
<td>$109,559</td>
</tr>
<tr>
<td>Ancient Tombs</td>
<td>6</td>
<td>$93,451</td>
</tr>
<tr>
<td>Mideast Studies</td>
<td>7</td>
<td>$69,780</td>
</tr>
<tr>
<td>Early Developments</td>
<td>8</td>
<td>$68,159</td>
</tr>
<tr>
<td>Religious Wars</td>
<td>9</td>
<td>$53,382</td>
</tr>
</tbody>
</table>
“Let’s look at this graphically.”
“Let’s take a look at their spending patterns month by month.”

This is an example of the incredibly fast response that OLAP tools provide management. Response is so fast they are able to “browse” aggregates from hundreds of thousands of records. In addition to the instant response, graphic display help them visualize business patterns that are otherwise hidden in their data.
Conclusion

The information retrieval needs of a large organization can become complex and diverse. Adopting an IT strategy that supports multiple layers of technology and data enables an information systems group to respond to those needs in a manner that is flexible and proactive without endangering the performance and security of mainframe production systems.

The development of an integrated data warehouse may become a cornerstone of such a strategy. The existence of a warehouse allows organizations to retain multiple years of historic data. Standard query/report tools can access that data but performance may be very slow. Even if performance is tolerable, special client tools are needed to provide management with the ability to browse historical data in a fast and easy manner. OLAP technologies appear to be the first affordable solution to this challenge.

OLAP technologies compliment existing query/report tools. They ‘top off’ the data warehouse in a manner that enables senior management to interact with one of their most valuable resources, information.

Suggested Readings


5. ________, "MDD: Database Reaches the Next Dimension." Database Programming Design, April 1995, pp.27-38


7. ________, "The Truth About OLAP." DBMS, August 1995, pps 40-46

