

# Oracle9i Application Server: Oracle Reports

*An Oracle White Paper  
February 2001*

# Oracle9i Application Server: Oracle Reports

Introduction.....	3
Product Overview .....	4
High end reporting made simple.....	4
Rapid Development.....	4
Single Report Execution - Multiple outputs .....	6
Globalization of Data Publishing .....	7
Report Versioning, Customization and Personalization.....	7
Centralized Scaleable Reports .....	9
Scaleable Load Management .....	9
Report Cache to Optimize Resources.....	11
Centralized Security.....	11
Secure Oracle Reports Server.....	11
What:.....	12
When: .....	12
How:.....	13
Open Environment .....	14
Open Database.....	14
Multiple Connection and Dynamic SQL.....	14
Open Access .....	14
Web Deployment - the ultimate thin client .....	15
Business to Business data exchange .....	16
Integrated Approach to Business Intelligence.....	18
Extend pre-packaged Oracle Applications .....	20
Conclusion.....	20

# Oracle9i Application Server: Oracle Reports

## INTRODUCTION

As the new century begins and business operations have become more global in scope, it is more apparent than ever before that timely access to corporate information can result in significant competitive advantage. Because of this, many companies are empowering more of their employees to become 'knowledge workers', with the ability to make decisions based on appropriate access to corporate information. Further-more as the nationalities and cultural backgrounds of those empowered to make decisions diversifies, the need to present the information in languages other than that of the parent organization is causing many reporting requirements to change the manner in which information is distributed. From the traditional fixed format, batch orientated, hard copy environment to one where ease of access and currency of the information is paramount.

To date, the majority of the reports on which a business depends for it's operational needs, are in the domain of the professionally managed Information Systems (IS) department and can be characterized by their size, complexity and large target audience (the combination of which dictate that they be run centrally on powerful servers, often in a batch mode environment). As such, they have the advantage of high throughput and central management, but run the risk of representing data that is only as up to date as the last batch run (particularly if the distribution mechanism is via hard copy). Further more, due to the mission critical nature of the reports themselves, the tools used are often older, more proven technologies that, while efficient, are inflexible and require significant effort to modify as the business requirement changes.

Conversely with the development of the desktop computing environment and the graphical user interface in particular, there has been a significant growth in the PC centric reporting tools to allow 'knowledge-workers' to access corporate information in a relatively simple manner. However as productive as these tools have become, it is the limitations of the client/server environment itself that has prevented most companies from taking advantage of these tools for their mission critical or operational reporting needs. The requirement to install (and maintain)

complex software on each machine within the user community dramatically increases the work load on the internal systems department, while the desktop centric nature of the tool generally restricts the size and complexity of the report output due to limited available processing power. Further-more, the individual user nature of the client/server model forces the re-execution of the report for each user who requires the information, significantly increasing the overall hardware requirements and hence cost.

The goal therefore is to implement an enterprise level reporting structure which takes advantage of the benefits of both paradigms, that is; the speed, scalability and manageability of a centralized reporting system, with the ease of development and 'on-demand' nature of desktop development tools. Oracle Reports, the latest generation of Oracle's premier Enterprise level reporting tool, answers the reporting needs of today's business environment by amalgamating the benefits of both desktop and server based reporting into a single unified reporting environment.

## **PRODUCT OVERVIEW**

Oracle Reports is a powerful Enterprise Reporting tool to build and publish high quality, dynamically generated Web reports. Through data-driven tables of contents, hyperlinks, and drill-down chart hyperlinks, Oracle Reports provides users with an easy route to the information they require.

Develop your Reports in Oracle iDS Reports Developer and use wizards and SQL to create complex reports like summary reports, matrix report, or tabular report.

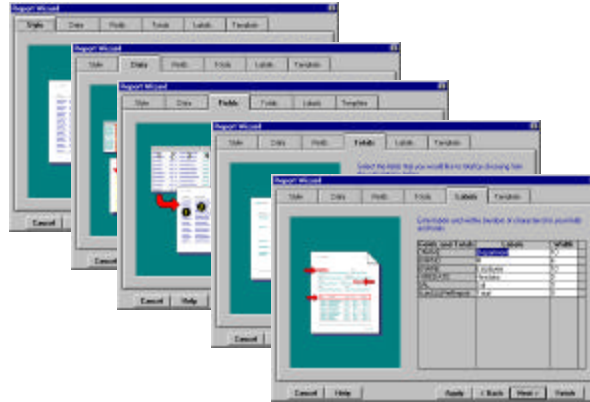
Publish Reports with Oracle9iAS Reports Services throughout your enterprise via a standard Web browser, and choose from many format options, including HTML, HTML Cascading Style Sheets (HTMLCSS), Adobe's Portable Document Format (PDF), XML, Rich Text Format (RTF), PostScript, PCL, or Delimited text. Schedule Reports for execution when you want them and for delivery by a method you choose, including e-mail or other means. Oracle Reports provides the most complete and comprehensive enterprise reporting environment available anywhere.

## **HIGH END REPORTING MADE SIMPLE**

### **Rapid Development**

Using a declarative, document centric development model, Oracle Reports Developer focuses a new user on the tasks required in the production of a report,

while still allowing experienced developers to have total control of their report development. Each major task in report production is expedited by the use of an appropriate Wizard, while the use of Report Templates and a live data pre-view allows for easy customization of the report structure. Once daunting tasks, such as complex data-models and layouts become simple 'drag & drop' actions. For example, the Query Wizard allows for the visual representation and construction of the complex SQL on which operational reports are often based, while the Report wizard steps the developer through the task of defining a report structure, data source, break groups and summaries.



**Figure 1: Re-Entrant Wizards Simplify Report Development**

Unlike many graphical reporting tools, Oracle Reports Developer does not apply any limitation on the number of queries defined within the report (many allow only a single query per report module), nor the overall format of the data returned. Based on page/frame model rather than the more simplistic banded reporting paradigm, Oracle Reports Developer allows for an almost unlimited number of report formats, without the need to resort to the complexity of nested report modules or complex procedural code.

On creation of a report definition, the report layout is displayed, along with the first page of data, in the 'Live Pre-Viewer'. In this WYSIWYG view the developer is able to see immediately the impact their format choices have, or will have, on the final report and hence, they can further enhance the output by the direct manipulation of report objects.

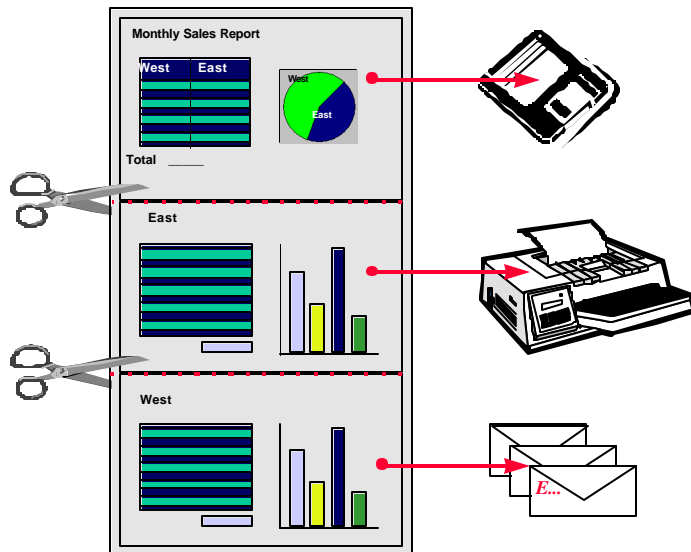
As the requirements for a report change with the changing business environment Oracle Reports Developer simplifies the modification process by use of re-entrant Wizard technology. That is, a developer may return repeatedly to a wizard to make further modification, without losing the current customizations that have been made. Currently most implementations of development wizards assume that any invocation is the first and will destroy any work to that point.

## Single Report Execution - Multiple outputs

When dealing with operational reports, there is often the need to produce multiple copies of a given report run, but to different output media. That is, a large report sent directly to the printer is also required in soft format for archival purposes. Likewise, in large organizations the user community may have differing requirements as far as the report delivery mechanism. A remote office, for example, may require their information via Email, while the local office prefers the physical printout.

This requirement to 'Burst' a report is one of the factors preventing many organizations from moving away from older 3GL based reporting tools and hence taking advantage of modern visual reporting environments. In the older procedural technologies, it was possible to open multiple output streams after each record fetch in order to generate the required multiple destination formats. The technique however, was code intensive, time consuming and required knowledge at design time of all the ultimate formats that would be needed.

Likewise, the needs of different target audiences often forces developers to produce several varieties of a given standard report. Each designed for an appropriate audience. For example, senior management may only require an executive summary of the data, whilst individual managers have need of the low level detail. In this case each report is querying the same data, it is only in the presentation of it that they differ. Unfortunately, this has generally resulted in the development of separate reports (each of which must perform the same or similar queries) or the use of complex procedural code to create the different reporting structures.



**Figure 2: Report Bursting to Multiple output Destinations and/or Formats**

Oracle Reports simplifies these issues in two ways;

1. Reports Developer allows for the definition of up to three separate sections within the report. Each section may have a completely different structure, page layout, page size, destination and output format, yet be based on the same data. At runtime each section may be selectively turned on or off to produce up to three different reports in a single run.
2. Reports Server permits multiple destinations from a single report run. By specifying the desired destinations at either the report or section levels, a single report execution can produce multiple copies of the output (or portions thereof) in different physical formats, whether it is to the printer, Email, File system or even the World Wide Web. This means therefore, that the developer is able to concentrate on the functionality of the report itself, rather than the mechanics of the destination format.

### **Globalization of Data Publishing**

As organizations become more global in scope there is an increasing need for a single corporate application to be used by the global audience. With the great diversity of languages with which a truly global application must deal, it has been a daunting task to 'internationalize' an organization's applications.

Oracle Reports Server helps organizations meet this challenge by supporting the publishing of data in different formats for an immense variety of languages and character-sets. In addition, the development environment itself of Oracle Reports Developer can be switched to reflect a developer's language preference (for a large number of languages).

### **Report Versioning, Customization and Personalization**

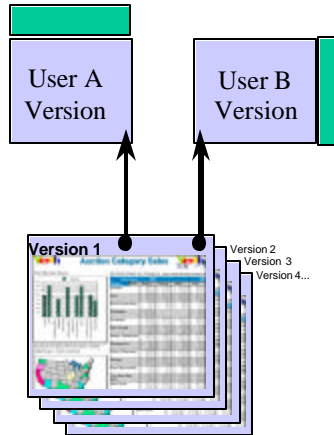
Though modern analysis and design techniques aim to meet the requirements of the entire user base as closely as possible, it is rare that a new report module will meet the needs of different users in quite the same manner. For example, the amount of information displayed may be insufficient for a given user or conversely superfluous to requirement, or simply in the wrong format. As such, it is often the case that the base report module is further customized for each unique need, resulting in a plethora of reports, each of which differs slightly from the original report definition

The problem is further compounded when a new version of the base report is released, as any custom modification would need to be rolled forward into the new version. This can result in a significant effort on the part of the application developers, and, in particular, greatly effects vendors of ERP style applications,

where the modules are often heavily modified for each customer site in order to meet their own corporate standards (e.g. Logo, Font, Look and Feel etc.)

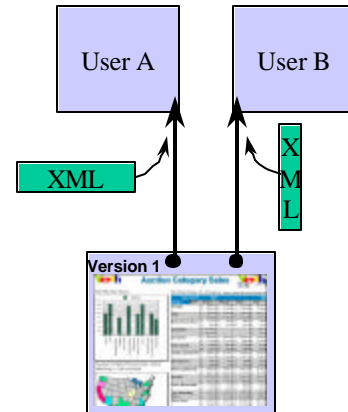
Oracle Reports Server answers this need by allowing the modifications to be externalized into a separate file, rather than having to create a unique version of the report. By creating and applying different customization files, the report output can be customized on a per user or group basis without changing the original report definition

Each user customizes the definition to create a **unique report** to meet their requirements.



Customization **must be re-applied** for each new version

A **single report** definition for all users.



Customization is **externalized** and applied at runtime.

### Figure 3 : Hard coded modifications Vs. Runtime customization with Oracle Reports

**Note:** In order for a customization file to be used as a stand alone report it must contain a specification for both a data model and layout. A partial definition contains sufficient information to modify an existing report definition (i.e. it can't run by itself)..

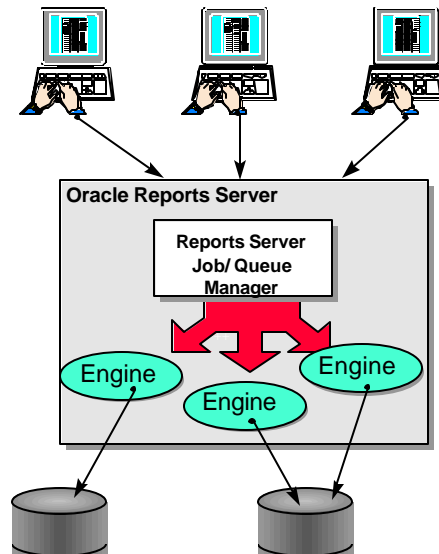
Defined using XML, the customization file allows for both the modification of current objects in the report and also the creation of new components. Using these XML tags it is possible to build a full or partial report definition that will then serve as either a customization file or even a completely self contained report.

The ability to externalize modifications greatly simplifies the upgrade process, as well as the need to make an application site specific. As an XML definition is applied, it is possible to save the resultant combined definition, resulting in a new unique module. By performing a batch update to the application's existing reports, it is possible to quickly update/upgrade a large number of modules without the need to open each file in the Reports Developer to manually make the changes. For example, a vendor of an ERP suite may wish to allow each customer to define the look and feel for their reports, requiring modification to fonts, colors etc. Through the use of XML based customization, gross changes of this nature can be made to the entire application in a single step.

## CENTRALIZED SCALEABLE REPORTS

### Scaleable Load Management

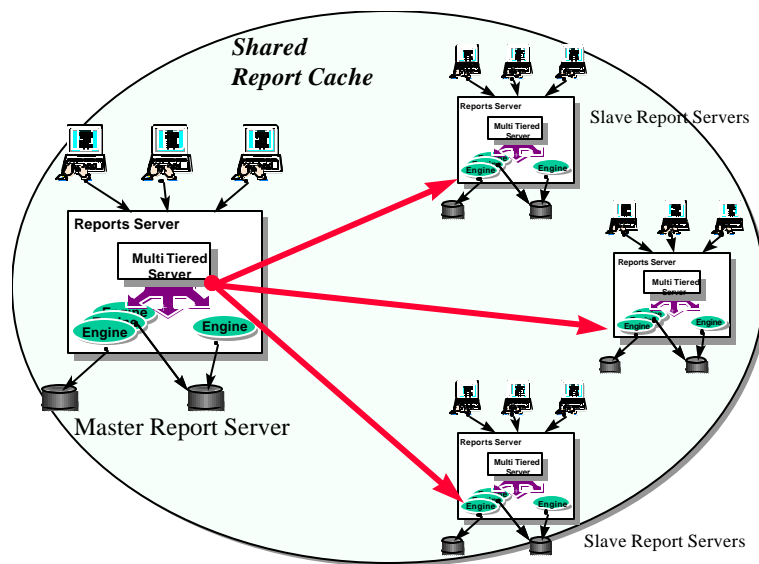
With the Reports Server it is possible to move the report execution engine to a third tier, thus the operational reporting infrastructure becomes an application service, to which distributed end users may subscribe from their client of choice. The Oracle Reports Server is a multi-process engine which is able to run multiple reports simultaneously, as requested by the distributed end users. Requests for report execution are entered into a job queue and dispatched in turn to a dynamic and configurable number of pre-spawned Oracle Reports runtime engines. As the number of jobs in the queue increases to the point where all available engines are active the Reports Server will dynamically invoke additional runtime engines as necessary to meet the increased load. Likewise as the peak load decreases and spawned engines become idle through insufficient job requests, the server will automatically shutdown idle runtime engines to conserve machine resources.



**Figure 4: Use of the Oracle Reports Server for distributed reporting**

If the user community is sufficiently large, there will be a point where, as new runtime engines are spawned, the processing power of a single computer is consumed and new report requests are forced to be queued rather than executed immediately. While the number of report requests at which this occurs is determined by the complexity of report, number of CPUs in the machine and operating system, there will always be a limit to the scalability that may be achieved from a single computer. In order to scale beyond the bounds of a single machine, Oracle Reports, in conjunction with the Oracle Reports Server, takes advantage of cooperative clustering to give almost infinite scalability.

By registering individual reports servers into a cluster (maintained through a master server), they are able to act as a single logical, but distributed, reporting engine. As jobs are submitted to the Report Server they are automatically re-routed to an available engine running on any machine within the cluster. In this manner, if the reporting needs of the user base is such that performance is degrading, it is simply a matter of defining a new machine to the cluster, to further extend the available processing power. As machines are added to the cluster the Report server will automatically recognize their existence and start to issue job requests to it. Conversely, if one of the machines becomes unavailable (due to failure or shutdown) the server will automatically cease to issue requests to that specific machine. The automatic 'Plug-n-play' nature of cooperatively clustered report servers means that scalability is limited by the total number of machines available, not by the processing power of any one computer.



**Figure 5: Cooperative Report Server Clustering for infinite scalability**

The ability to perform dynamic load management answers one of the greatest issues with a centralized model, that is, how to size the server itself when sharing the available hardware. By configuring itself to the current peak load the Oracle Reports Server maximizes the resources available for other running processes on the system while retaining the performance required to get timely information. Furthermore, it is often the case, that due to administrative or performance constraints a report needs to be run at a time other than when it was submitted. That is, asynchronously. The Oracle Reports Server enables administrators to further maximize the available processing power by automatically scheduling report execution at a more convenient time, such as when overall system usage is low (e.g. the middle of the night).

## **Report Cache to Optimize Resources**

When the activity of a large user community is primarily the running of an application suite, the likelihood of multiple users issuing a duplicate request for the same report is quite high. In this case, dynamic generation (as with the client server model) would result in unnecessary load on the application system as the same data is to be queried and subsequently formatted for each request.

To further maximize resource availability on the application server the Oracle Reports Server provides the ability to cache generated report output. If, within a defined time period, a duplicate request is received, it is automatically serviced from the cache rather than by re-execution of the report. Hence, there is both improved performance and resource availability due to the fact that there is no subsequent database access or application execution.

## **CENTRALIZED SECURITY**

It is often the case that user authorization is defined not only by the privileges on the data itself (that is, database object level privilege) but also the functions performed upon it.

In it's simplest form, an authorization scheme designed to enforce application level security (one based on the right of the user to access the function as well as the data) should allow for the definition of the following criteria:

- Who has the appropriate privileges to access the defined function or object.
- What business functions and objects are available to be accessed.
- When may the defined function or object be accessed. That is, is the availability determined by either organizational or procedural controls (e.g. only available during business hours or after the 'close of books' in a financial period).
- How may the function be implemented by the user. That is, given the authorization to access given function the user may only specify certain parameters to that function (e.g. may only specify output formats or printers).

## **Secure Oracle Reports Server**

By integration with the Oracle Portal security repository, the Oracle Reports Server is able control user access to the reports being executed on the Server. As a user makes a request for a given report, the Oracle Reports Server uses the information stored within it's security repository to verify their access rights. That is, is the user allowed to run the requested report, at that time, and on the requested server.

The access rights exposed by the Oracle Report Server cover the following areas:

**What:**

As a given report may reside on more than one Reports Server the definition of a user's access privileges must go beyond the report itself to include the available hardware on which that report may be executed/output. For example, a given report resides on both the central Reports Server and the one used by senior management. Even though a given user may have the right to run the report, he/she should be prevented from executing it on the management server. As such, the Oracle Reports Security allows for the securing of a number of different objects relating to the execution of a report.

**Printer Access:**

- What local or network printers are available.
- Who has the appropriate privileges to use a given printer.
- When is this printer available for printing reports.

**Server Access:**

- What Report Servers are available to accept job requests.
- What Printers are available on a given Reports Server.
- Who has the appropriate privileges to submit job requests to a given Reports Server.
- When is a given Reports Server available to accept job requests.

**Report Module Access**

- What Oracle Reports have been registered for secure access.
- Who has the appropriate privileges to run a given report.
- Which Reports Servers have a given report available.
- When is a given report available for execution.

**When:**

It is often the case that a user's ability to access a report, server, or printer is determined by the point in time the job was submitted to execute. With the universal access afforded by the web, the simple physical security of not being able to access the appropriate hardware (PC, Printer etc.) outside of a defined time period is removed, hence the need to define an availability window as part of the report/object definition becomes paramount. For example, it has been decided that a given report may only be accessed during standard business hours. By applying an availability calendar which defines standard business hours

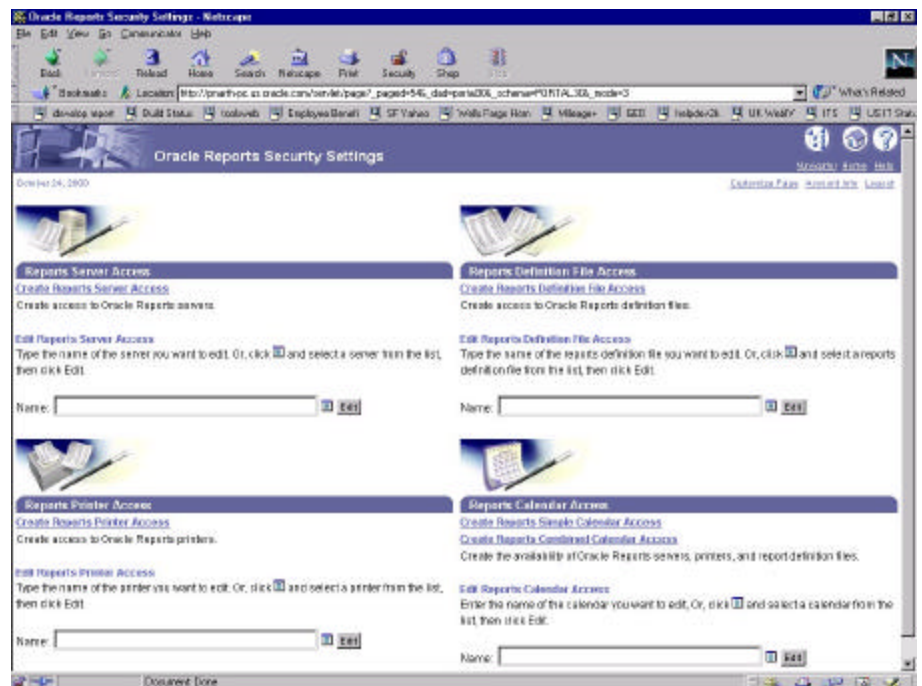
employees would not be able to log-in from home after hours and access the report.

The Oracle Reports Server supports the ability to define availability calendars for either:

- The individual report.
- The Reports Server on which jobs may be executed.
- The printers to which the report output may be sent.

**How:**

It is often important not only to specify which report a user may run (and when), but also how that user may interact with a given report. The Oracle Reports Server achieves this by restricting the report request options (i.e. required and optional report parameters), which are exposed to a given user at job submission. By specifying which parameters are exposed, the administrator can allow different users or groups to apply different options to the same physical report definition file.



**Figure 6: Web based Reports Security Administration**

In order to define the access privileges mentioned, the Oracle Reports Server takes advantage of the web based interface of Oracle Portal. That is, through the use of a browser and a simple wizard style interface, the system administrator may

implement a security infrastructure, for a distributed reporting environment, from a single console.

## **OPEN ENVIRONMENT**

### **Open Database**

Oracle Reports and the Oracle Reports Server were designed to be truly open, not only from the standpoint of the data source, but also from the choice of thin client from which to submit report requests. Oracle Reports, like other related tools from Oracle, is able to access foreign data-stores such as SQL-Server, Informix and Rdb via the use of either optimized ODBC or native connections. As such, reports developed using Oracle Reports are able to seamlessly integrate with much of the functionality found within those data-sources including the use of store procedures to return values used within report calculations.

### **Multiple Connection and Dynamic SQL**

Whereas most organizations have now tended to standardize on a single database (predominantly Oracle), due to fact that many departments have made separate purchasing decisions in the past (with a need to support a legacy systems etc.), it is common that a given company will be running multiple databases from multiple vendors. As such, the resulting 'islands of information' have made it difficult for a report to access all of a organization's data in a single run. That is, the report execution is driven through a connection to a single database.

Oracle Reports Server is able to join these 'islands of information' into a single report by allowing for the creation of any number of simultaneous database connections during the report run. That is, while the Oracle Reports Server requires a connection to a primary data source to retrieve the main mass of data for publication, use of the EXEC\_SQL function allows for the ability to open subsequent, and separate, connection handles to other databases in order to return the small 'islands' of related information.

As well supporting multiple database connections, use of EXEC\_SQL allows for the dynamic creation of the SQL statement used to return the required information. That is, for a given report run the SQL statement may be created on the fly then immediately executed to return data.

### **Open Access**

With a widely distributed user base that may have greatly differing application environments, the need to have multiple methods to access the reporting service is a great imperative. As such the Oracle Reports Server allows the end user to submit a request from the client environment of choice.

The predominance of Microsoft Windows on the desktop has lead to a growth in the market for work group orientated visual development tools (such as Visual

**Note: This function may also be used to open separate connections to the Oracle Database itself as well as to foreign data-stores.**

Basic and Delphi) that are designed to run on that platform. However, due in part to their emphasis on screen development, there has been little ability to access the mission critical reporting environment of the greater organization. In order to address this limitation, Oracle Reports includes a lightweight ActiveX based interface to the Oracle Reports Server. Therefore, any development tool that supports the use of such a component may open a conversation with a remote server and submit requests for a report.

In the same vane, end-users would like to incorporate the output from their corporate reports into other projects developed through desktop productivity tools such as word processors and spreadsheets. Oracle Reports, meets this requirement by supporting direct export to a number of common import formats such as Rich Text Format (word processors) and Comma/Tab delimited output (Spreadsheets).

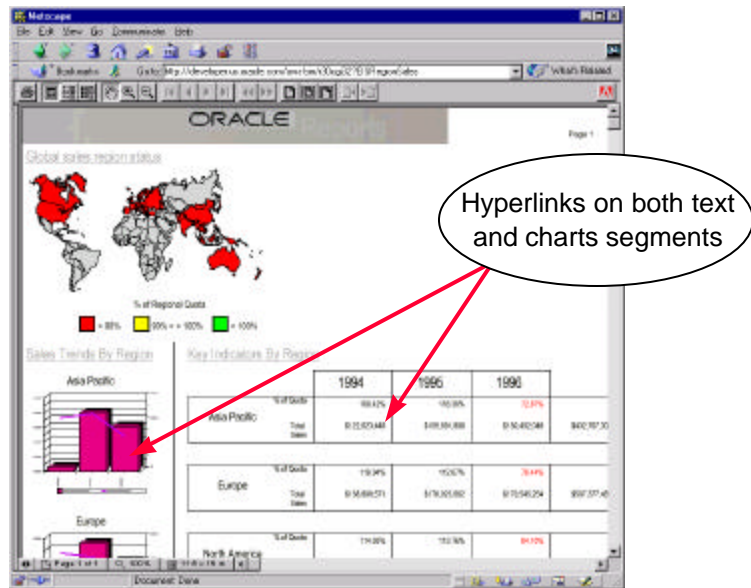
If the application developer is using Oracle Forms, they may take advantage of a native server interface that allows for a direct submission of report requests, or if these interfaces do not meet the developer's requirements, a lightweight thin client allows for a simple command line interface.

#### **WEB DEPLOYMENT - THE ULTIMATE THIN CLIENT**

It was the need to share information throughout their user community that lead the Physicists and Engineers at CERN to develop the technology that has evolved into what is now known as the World Wide Web. The ability to publish a document in Hyper Text Markup Language (HTML) allowed for a single copy of a document to be accessed by a widely distributed group of users, simply by use of a light-weight browser.

In concert with the Oracle Reports Server, Oracle Reports delivers a complete database publishing environment by providing the ability to generate and distribute reports dynamically to both a company Intranet and the Internet. Through the use of a web server interface and subsequent on-demand generation of either HTML or Adobe's Portable Document Format (for extremely high fidelity electronic documents) the ability for users to access up-to the minute database information on the web is now a reality. In order to allow for qualification of the information required, Oracle Reports allows for the parameterization and definition of criteria at runtime. However, as the definition of information on the web is via a Uniform Resource Location (URL) it would require the 'knowledge-worker' to know all possible parameters to all available URLs. Hence to allow the browser to truly become an information client, the Oracle Reports Server will automatically generate at runtime an HTML based parameter form in which to display and subsequently define the required parameters. This, along with the ability to dynamically drill-down into the data (via Hyper-links) takes the web from

a simple information publishing tool to one where real-time analysis of data is now possible.



**Figure 7: Drill down to detailed information**

The process of integration with the web server is simplified via the use of the Oracle Reports Server web interface component which supports the Common Gateway Interface (CGI), and the use of Java Servlets. With this interface web server administrators will now be able to publish database information directly to their user base, via the definition of a URL, without the need to resort to more complex (and error prone) methods such as Perl and CGI scripting.

Through use of a simple web browser, the Internet and the Oracle Reports Server, 'knowledge-workers' may now be anywhere on the planet and receive the latest information on demand.

### **BUSINESS TO BUSINESS DATA EXCHANGE**

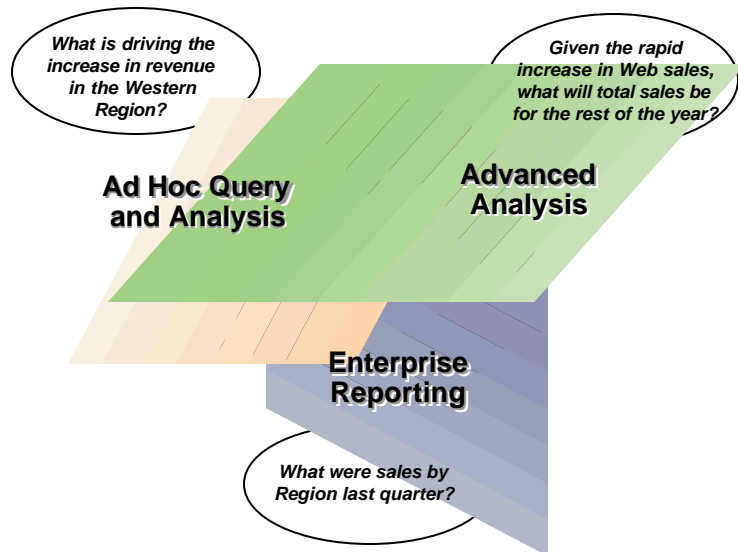
In the past the ability for a business organization to directly share its data with another, such as a supplier or financial institution, was tied to the form of Electronic Data Interchange (EDI) chosen by both parties. The format required for EDI used was often industry/application specific and in many cases was proprietary in nature. Further-more the actual method of data transfer generally involved physically writing the information to a removable media (such as tape) and sending it to the receiving organization to be read. The inherent complexity and manual input of this form of EDI limited its use to those applications where immediate access to current data was not required (such as sending financial information after month end accounting) and where the cost to the organization was warranted.

As the usage of the Internet expands and more organizations go 'on-line' to implement an 'eBusiness' strategy, the need to share information with partner companies in a timely manner becomes more imperative. For example, an eStore web site requires the current catalogue and stock on hand from a given supplier or suppliers, and it may have to jump between several suppliers in order to meet its customers needs. For a on-line business the time and effort required to implement a traditional form of EDI could result in the presentation out of date information, dissatisfied customers and hence lost revenue.

What is required is a method that would allow an eBusiness to easily access to an external organization's data via the Internet. In this environment allowing direct SQL access (as would often be the case in an Intranet application) is not a feasible option, as it would both raise significant security issues for the owner of the data, as well as require each web site accessing the information to have in depth knowledge of the available data structures. As such, a solution that allows a supplier to expose specific information in a simple/open manner (without having to predefine the structure/mechanism) is needed. With the development of the eXtensible Mark-up Language (XML) this has becomes possible. By encapsulating the required data within an XML document, a supplier can expose the information to the web in an industry standard format, which may then be parsed directly by the calling application.

The Oracle Reports Server simplifies this data transfer by allowing for the automatic generation of the XML file. As a URL is all that is required to invoke an Oracle Report on the Web, by the definition of a Report module to query the data, the supplier can stream the information back to the calling eCommerce application in real time. Without the need to know anything about the underlying XML syntax or technology.

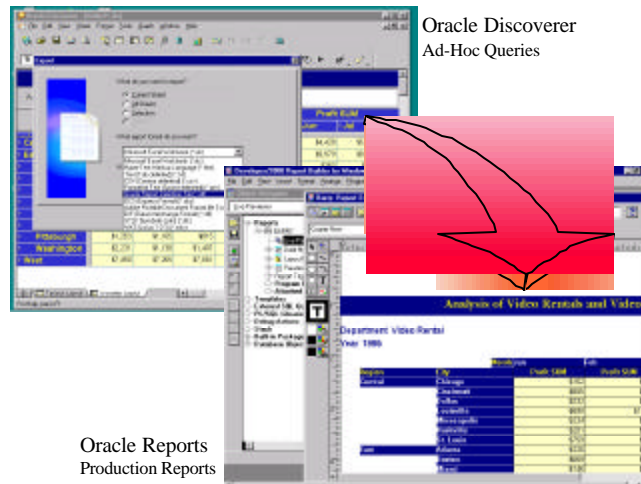




**Figure 9: Integrated business Intelligence across the enterprise**

In order to make the most of the information gained from analytical tools it needs to be accessible to as wide an audience as possible. Oracle Reports facilitates the publishing of analytical data to the enterprise in two ways;

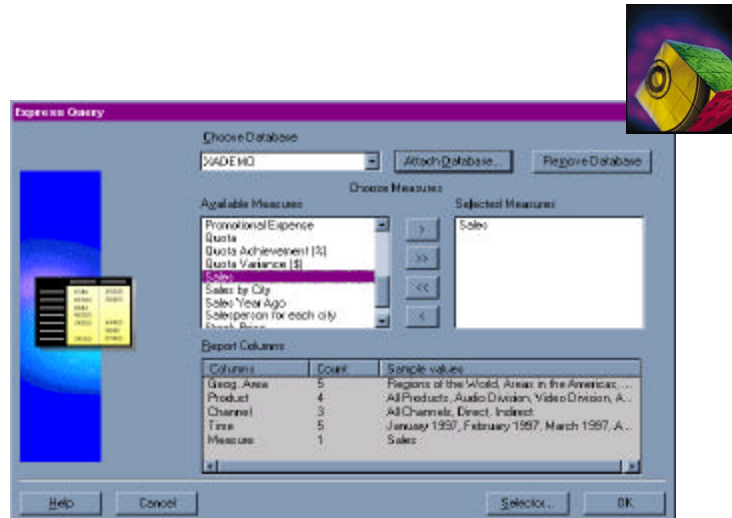
1. By allowing for the creation of reports directly from dataviews created in Oracle Discoverer, Oracle's award winning ad-hoc query and analysis tool for Relational databases.



**Figure 10: Generate Operational Reports directly from Ad-Hoc queries**

2. The creation of Express queries directly against the Oracle Express Server, an advanced calculation engine and multi-dimensional cache for on-line analytical processing (OLAP). By the use of the Express

Wizard within the Oracle Reports Developer query definition screen, users may choose from lists the dimensions and measures which have been defined within the multi-dimensional cache.



**Figure 11: Build Reports directly against Multi-Dimensional Data**

Note: Oracle Applications Release 11i was built with Oracle Reports Release 6i.

### EXTEND PRE-PACKAGED ORACLE APPLICATIONS

Oracle Corporation is the second largest vendor of pre-packaged applications. With Oracle Reports Developer you can extend and enhance Oracle's pre-packaged applications, tailoring them to the unique needs of your organization.

### CONCLUSION

Oracle iDS Reports Developer in conjunction with the Oracle9iAS Reports Server brings a new level of flexibility to enterprise level reporting by merging the benefits of both server and desktop centric reporting environments. With an easy to use, highly productive development environment organizations may react quickly to the changing reporting needs of their business and user community while at the same time gaining the benefits of central server performance, security and management.



Oracle9i Application Server: Oracle Reports  
February 2001  
Author: Barry Hiern  
Contributing Author: Paul Narth

Oracle Corporation  
World Headquarters  
500 Oracle Parkway  
Redwood Shores, CA 94065  
U.S.A.

Worldwide Inquiries:  
Phone: +1.650.506.7000  
Fax: +1.650.506.7200  
[www.oracle.com](http://www.oracle.com)

Oracle Corporation provides the software  
that powers the internet.

Oracle is a registered trademark of Oracle Corporation. Various  
product and service names referenced herein may be trademarks  
of Oracle Corporation. All other product and service names  
mentioned may be trademarks of their respective owners.

Copyright © 2001 Oracle Corporation  
All rights reserved.