

Security and the Data Warehouse

An Oracle White Paper
April 2005

Security and the Data Warehouse

Executive Overview	2
Why is security important for a data warehouse?	2
Oracle's strategy for data warehouse security	3
Consolidation as an enabler of security	4
Data Warehouse Security.....	5
Controlling Access to Warehouse Data.....	5
Role-based Access Control	6
Row-Level Security	6
Virtual Private Database.....	7
How It Works	7
Query Performance	8
Virtual Private Database in Oracle Database 10g	9
Column-Level Virtual Private Database.....	9
Policy Types for Added Performance	10
Benefits for the Data Warehouse	10
Oracle Label Security.....	11
How It Works	11
Label Benefits for the Data Warehouse	12
Additional Protection With Encryption.....	12
User Accountability	13
Auditing the Data Warehouse.....	13
Fine-grained Auditing for Data Warehouses	13
Benefits of Auditing the Data Warehouse	14
Customer Examples	14
Case Study #1: Data Warehouse for a Government	14
Case Study #2: A Data Syndicator	15
Case study #3: A major financial institution.....	15
Summary	16

EXECUTIVE OVERVIEW

Data warehousing poses its own set of challenges for security: enterprise data warehouses are often very large systems, serving many user communities with varying security needs, and while data warehouses require a flexible and powerful security infrastructure, the security capabilities must seamlessly operate in an environment which has stringent performance and scalability requirements. Security must be built into the core of a data warehouse.

WHY IS SECURITY IMPORTANT FOR A DATA WAREHOUSE?

Many of the basic requirements for security are well-known, and apply equally to a data warehouse as to any other system: The application must prevent unauthorized users from accessing or modifying data, the applications and underlying data must not be susceptible to data-theft by hackers, the data must be available to the right users at the right time, and the system must keep a record of activities performed by its users. These requirements are perhaps even more important in a data warehouse because by definition a data warehouse contains data consolidated from multiple sources, and thus from the perspective of a malicious individual trying to steal information a data warehouse can be one of the most lucrative targets in an enterprise.

However, beyond these fundamental and obligatory requirements, there are additional scenarios in which a robust security infrastructure can vastly improve the effectiveness or reduce the costs of a data warehouse environment.

Some typical customer scenarios for data warehousing security include:

- A company is managing an enterprise data warehouse that will be widely used by many divisions and subsidiaries. That company needs a security infrastructure that ensures that the employees of each division to only be able to view only the data that is relevant to their own division, while also allowing for employees in its corporate offices to view the overall picture.
- A company's data warehouses stores personal information. Privacy laws may govern the use of such personal information. The adherence to these privacy laws must be implemented in the data warehouse.
- A company sells data from a data warehouse to its clients. Those clients may only view the data to which they have purchased or subscribed; they

should never be permitted to see the data of other clients (especially since those other clients may be competitors).

ORACLE'S STRATEGY FOR DATA WAREHOUSE SECURITY

How should the data warehouse team approach security? There are several key considerations when implementing a data warehouse.

The first consideration, which is hardly surprising, is that the data warehouse team must consider end-to-end security. A data warehouse environment consists of much more than just a database. The entire environment ranges from the extraction of data from operational system, transportation of this data to the data warehouse, the possible distribution of this data to data marts and other analytic servers, and finally the dissemination of this data to end-users. The environment spans multiple servers and multiple software products ... and of course every component needs to be secure.

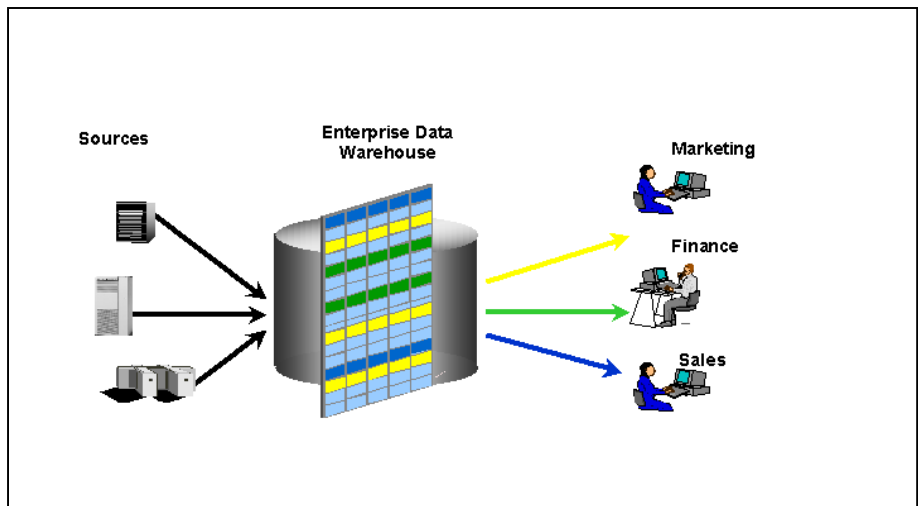


Figure 1: Consolidation in an enterprise data warehouse with built-in security.

There are undoubtedly many data warehouses today in which the database itself has little risk of a security breach, but at the same time the flat files which are used to populate the data warehouse are stored in an unsecured location. This is an example of the security loopholes that can emerge when the entire data-warehouse process has not been designed with security in mind.

The second consideration is related to the interaction of security and the data warehouse architecture. A consolidated data warehouse is much simpler to secure than dozens of heterogeneous data marts. Indeed, many industry analysts and customers agree that an enterprise data warehouse is the preferred implementation model, and among that model's many virtues is the fact that a centralized data warehouse's security is simpler and less expensive to manage, while providing higher levels of security.

The final consideration is the recognition the core of a data warehouse is the data. Although end-to-end security is crucial, the ability to provide a flexible multi-layer security model on the data in the data warehouse is nevertheless the primary requirement for data warehouse security.

This white paper will discuss the latter two considerations in detail.

Consolidation as an enabler of security

Many analysts have described consolidation as a key enabler of a successful enterprise data warehouse architecture. While some companies in the past have built multiple data marts, the disadvantage of a data-mart approach is that different organizations are viewing different sets of data, and may be calculating different answers to the same questions. The risks of making decisions based on inconsistent data are considerable. Data marts have a further shortcoming, in that an environment of multiple, heterogeneous systems is expensive and unwieldy to manage. A single enterprise data warehouse addresses both the business and technical shortcomings of disparate data marts.

Formerly, when information was entered into a business system, it was often compartmentalized. Information maintained by each internal department, such as sales, manufacturing, distribution, and finance, was kept separate, and was often processed by physically separate and incompatible databases and applications. This prevented businesses from taking full advantage of the information they already had, since it was difficult for different departments to exchange information when it was needed, or for executives to get the latest and most accurate “big picture” of the business. Companies have found that linking disparate data sources and consolidating them where possible, allows users to obtain better information, and to get more benefit from that information, which thus makes the information more valuable.

This consolidated data warehouse model also provides many benefits for data warehouse security:

- **Consistent security model:** If you have dozens of data marts, based upon different database servers, it may be difficult if not impossible to implement consistent security policies identically across all of those data-marts. With a single data warehouse, you implement the security policies in one place and they are applied consistently across all of the data.
- **Central, expert management:** With a central data warehouse, an organization can devote more resources towards providing sound security. Security is simply one part of administration. It is doubtful, or at least almost prohibitively expensive, for an enterprise to implement the same level of security across multiple data marts than it can implement on a centrally managed data warehouse.
- **Fewer points of attack:** When data is spread among dozens of data marts, a malicious employee can choose the weakest system to attack. With

a consolidated system, there is only a single system that needs to be secured.

- **Simpler security maintenance:** Many well-publicized security breaches have been caused by the simple fact that patches have not been applied to all systems before someone tried to hack one of them. A single data warehouse allows rapid installation of patches and is much simpler to administer.

In short, the benefits of a consolidated data warehouse (consistent data and simpler administration) apply to all aspects of data warehousing, and especially to security.

There is a potential trade-off to consolidation. Improving the value of data available to legitimate users generally improves its value to intruders as well, increasing the potential rewards to be gained from unauthorized access to that data, and the potential damage that can be done to the business if the data were corrupted. In other words, the more effective a data warehouse is, the greater the need to protect it against unauthorized access.

DATA WAREHOUSE SECURITY

Among the best ways to mitigate security risk is to provide multiple layers of security mechanisms, so that failure of a single mechanism does not result in compromise of critical information. Oracle Database 10g addresses data warehouse security, performance and scalability objectives by enabling consolidation and layering database-enforced security that cannot be bypassed.

Controlling Access to Warehouse Data

A data warehouse must ensure that sensitive data does not fall into the wrong hands, and this is especially important when the data is consolidated into one large data warehouse. Upon creating a database user and granting him or her the rights to connect to the data warehouse, the administrator who manages the data warehouse must control access to data, and they often must limit a particular user's access to the level of individual records in a database table based on the identity and privilege of that user.

Organizations struggle with the implementation of this type of strict, granular access control by building application code in each of the front-end applications. Maintaining this type of complex access control code is not only costly, but also risk-prone. If access control is built into an application, but a user has access to the database itself, which is common in warehouse environments—through SQL*Plus or a reporting tool—then the application logic and access control can be bypassed. For these principal reasons, organizations that understand this hard-to-solve problem build security as a whole, and access control in particular, onto the data itself, inside the data warehouse.

Role-based Access Control

Database privileges and roles ensure that a user can only perform an operation on a database object if that he or she has been authorized to perform that operation. A privilege is an authorization to perform a particular operation; without explicitly granted privileges, a user cannot access any information in the database.

System privileges authorize a user to perform a specific operation, such as the CREATE TABLE privilege, which allows a user to create a database table. Object privileges authorize a user to perform a specific operation on a particular object. An example of object privileges is SELECT ON SALES_HISTORY, to allow a particular user to query to the fact table, but not query other database objects, nor modify any of them. Granting a user SELECT, UPDATE, INSERT ON EMEA_SALES allow a user to read and write to this view.

By providing these types of privileges, the Oracle database enables you to ensure that database users are only authorized to perform those specific operations required by their job functions. In addition, other features, such as roles and stored procedures, not only allow you to control which privileges a user has, but under what conditions he can use those privileges.

While privileges let you restrict the types of operations a user can perform, managing these privileges may be complex. To address the complexity of privilege management, database roles encapsulate one or more privileges that can be granted to and revoked from users. For example, you can create the PROMO_ANALYST role, grant it all privileges necessary for marketing promotion analysts to perform their jobs, and then simply grant this single role to all marketing analysts. In addition, you can create the PROMO_MANAGER role, grant it the PROMO_ANALYST role and any other necessary privileges, and then grant it to all marketing managers. To later grant or revoke an additional privilege to all of these users, you need only grant or revoke that privilege to the PROMO_ANALYST role.

Roles and privileges, or Role-based Access Control (RBAC), enforce security on the data itself, and their use is essential to a data warehouse because users access data via a number of applications and tools. Roles enforce object-level security, which can be enhanced with the enforcement of security at the level of individual rows within a database object.

Row-Level Security

Oracle Database 10g sets the standard in data warehouse security with support for row-level security (RLS), which is unique to the Oracle database. Oracle's Virtual Private Database feature, and the Oracle Label Security technology derived from it, enforces row-level security on database tables, views, materialized views, and synonyms. These two technologies are key enablers to consolidating data into one large data warehouse.

Virtual Private Database

Virtual Private Database (VPD) is server-enforced, fine-grained access control, together with secure application context. By dynamically appending SQL statements with a predicate, VPD limits access to data at the row level and ties a security policy to the database object itself (specifically, a table, view, or synonym). It enables multiple users to have secure direct access to mission-critical data within a single database server, with the assurance of complete data separation. Virtual Private Database can ensure that banking customers see only their own account history, and that a company serving multiple companies' data (who may be competitors to one another) can do so from the same data warehouse, and allow each company to see only its own data.

Virtual Private Database is application transparent. Security is enforced at the database layer and takes into account application-specific logic used to limit data access within the database. Both commercial off-the-shelf applications and custom-built applications can take advantage of its granular access control, without the need to change a single line of application code.

Within an enterprise, the Virtual Private Database results in lower cost of ownership in deploying applications. Security can be built once, in the warehouse, rather than in every application that accesses data. Security is stronger, because it is enforced by the database, no matter how a user accesses data. Security cannot be bypassed by a user accessing data via an ad hoc query tool or new report writer. In an enterprise data warehouse, which often supports dozens of different applications as well as many end-user tools, Virtual Private Database is key enabling technology.

How It Works

Virtual Private Database is enabled by associating a security “policy” with a table, view, or synonym. An administrator uses the supplied PL/SQL package, DBMS_RLS, to bind a policy function with a database object. Direct or indirect access to the object with an attached security policy causes the database to consult a function implementing the policy. The policy function returns a predicate (a WHERE clause) which the database appends to the user's SQL statement, thus *transparently and dynamically* modifying the user's data access.

An “application context” enables access conditions to be based on virtually any attribute an administrator deems significant, such as organization, subscriber number, account number, or position. For example, a warehouse of sales data can enforce access based on customer number, and whether the user is a customer, a sales representative or a marketing analyst. In this way, customers can view their order history over the web (but only for their own orders), while sales representatives can view multiple orders, but only for their own customers, and analysts can analyze all sales from the previous two quarters.

Application contexts act as a secure cache of data that may be applied to a fine-grained access control policy on a particular object. Upon user login to the database, Oracle sets up an application context to cache information in the user's

session. Information in the application context is defined by a developer based on information relevant to the particular application. For example, a reporting application that will query Regional Sales data can base its access control on the user's position and division. The application, in this case, could initially set up an application context for each user as he/she logs in and populate it with data queried from the Employees and Departments tables for the user's position and division, respectively. The package implementing the VPD policy on the Regional Sales table references this application context to populate the user's position and division for each query. As such, Application Context obviates the need to execute sub-queries, which might otherwise hinder performance.

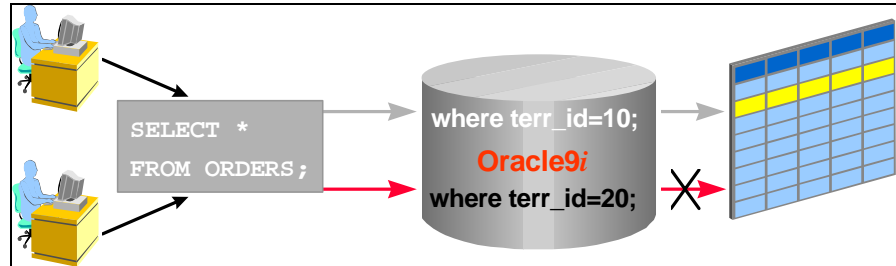


Figure 2: Virtual Private Database dynamically and transparently modifies SQL with a security predicate, enforcing a non-bypassable security policy.

Query Performance

One important consideration for all data warehouse administrators is query performance. Thus, one key question is how Virtual Private Database impacts query performance. The answer is that VPD has almost no impact on query performance and in fact in many data warehouse environments, VPD provides better query performance by virtue of the fact that queries are returning less data due to the security policy. These performance characteristics are perhaps one of the biggest strengths of Virtual Private Database for a data warehouse environment.

The reason that VPD provides such good query performance is that VPD operates by adding predicates to an end-user's SQL statement. That is, if an end-user issues a query such as:

```
select t.month, sum(s.sales_amount)
from sales s , times t
where s.time_id = t.time_id
and t.year = 2002;
```

Then VPD might modify that query as follows:

```
select t.month, sum(s.sales_amount)
from sales s , times t
where s.time_id = t.time_id
```

```
and t.year = 2002
and s.product_id in (100, 101, 102);
```

The query with the Virtual Private Database policy applied is still a standard SQL query. All performance optimizations occur after the VPD policy is applied, so that the VPD-modified SQL will fully leverage all of Oracle's query processing technology, such as indexes, partitioning, materialized views, and parallelism. This is why Virtual Private Database does not significantly affect query performance, even when VPD policies are applied to tables with billions of records. In fact, the new predicate(s) introduced by the VPD policy may enable Oracle to further improve query performance by utilizing additional indexes or leveraging partition-pruning.

Virtual Private Database in Oracle Database 10g

Virtual Private Database is available in the Enterprise Edition of Oracle8i Database, Oracle9i Database, and Oracle Database 10g. Being that the Oracle Database 10g release is the third version of this feature set, it provides significant, rich features. Column-Level VPD policies provide stricter access controls on data stored in data warehouses. Static and Context-Sensitive policies optimize VPD for significant performance improvements.

Column-Level Virtual Private Database

In Oracle Database 10g, VPD policies can be applied only where a particular column or columns are accessed in the user's query or DML statement. This means that when a user has rights to access the object itself (with an object-level privilege), VPD can limit the individual rows returned only if the columns he or she accesses are considered security-relevant.

For example, a data warehouse containing medical records with a fact table containing tens of thousands of records of hospital stays can apply a VPD policy with the "relevant_columns" of Social Security Number (SSN) and DIAGNOSIS. When a physician queries patient names and their hospital admission dates, the Oracle database allows her to see all patients, as their names and admission dates are not deemed to be security-sensitive columns for physicians who have SELECT privilege on this table. For this query, VPD does not append the SQL with a where clause. However, if the physician queries patient names and their diagnoses, VPD dynamically rewrites the query with "where physician_name = 'sarah_carsen'" to enforce that the doctor sees only her only patients' diagnoses.

Column-level VPD provides added security and privacy to the data warehouse by allowing an administrator to flag particularly security- or privacy-sensitive data stored in columns.

Policy Types for Added Performance

VPD policies in Oracle8*i* and Oracle9*i* are dynamic; that is, the database executes the policy function upon every query or DML. In addition to dynamic policies, Oracle Database 10g provides the “static” and “context-sensitive” VPD policies. These policy types provide added performance improvements through optimization.

A static policy maintains the same predicate (where application context changes the value) for queries, updates, inserts and deletes throughout a session. Static policies are particularly useful for hosting environments in which you always need to apply the same predicate. For these situations, it would add unnecessary overhead to the system to re-execute the policy function for every SQL, when the policy function will always append the same predicate. As an example, take a data warehouse that contain market research data for customer organizations who are competitors to one another. The warehouse must always enforce that an organization sees its market research data only, expressed by the VPD predicate “where subscriber_id = sys_context('ASP', 'company_id')”. The use of sys_context within the application context enables the database to dynamically change which organization’s rows should be returned. There is no need to re-execute the policy function, which will result in the same predicate for every query, so the static policy caches the predicate to optimize the SQL.

Virtual Private Database also supports context-sensitive policies. Here, the VPD predicate can change after statement parse time. Only if the application context has changed will VPD re-execute the policy function (at execute time) to ensure it captures any changes to the predicate since the initial parse. This type of policy is especially useful when VPD policies must enforce two or more different predicates for different users or groups. A SALES_HISTORY table with a single policy of “analysts see only their own products and regional employees see only their own region” needs to execute the policy function each time the user changes.

Additionally, both static policies and context-sensitive ones can be shared across multiple database objects, so that queries on another database object can use the same cached predicate. Like static policies, context-sensitive policies save the overhead of re-executing the policy function upon every query, significantly reducing any performance impact.

Benefits for the Data Warehouse

Virtual Private Database for the data warehouse ensures that, no matter how a user gets to the data—through any application or tool—the same strong access control policy is enforced. Security is built-in to the warehouse, so there is no channel in which users, even those with direct SQL access, can bypass that security. Because the Oracle database natively supports VPD, it automatically leverages all of the query processing and optimization features of the RDBMS for performance. Virtual Private Database scales to data warehouse environments, even where the tables have billions of rows. In sum, data warehouses with Virtual Private Database

can centralize data from multiple sources in one large warehouse, with benefits of central administration, and the protection of strictly-enforced data separation. No other technology is proven to enable the secure data warehouses more than Virtual Private Database.

Oracle Label Security

Oracle Label Security, a security option for the Oracle Database, extends Virtual Private Database to enforce label-based access control. Oracle Label Security is a complete VPD-enabled application, augmenting VPD with labeled data management. It increases the ease of deploying secure data warehouses and provides row level security out-of-the-box.

Label-based access control allows you to assign sensitivity labels to rows in a table, control access to that data based on those labels, and ensure that data is marked with the appropriate sensitivity label. For example, an organization may differentiate between “Company Confidential” information and “Partner” information. Further, there may be some “Company Confidential” information that can be shared with certain key partners, and some that is only accessible by certain subsets of internal groups, such as the Finance or Sales divisions. The ability to natively manage labeled data is a great advantage for organizations to provide proper information to appropriate people at the proper data access level.

How It Works

Oracle Label Security uses policies, which are collections of labels, user authorizations and security enforcement options. Once created, policies can be applied to entire application schemas or specific application tables. Oracle Label Security supports multiple policy definitions within a single warehouse. Label definitions, user authorizations and enforcement options are defined on a per-policy basis. For example, a Marketing policy might have labels such as Marketing-Only, Manager, and Senior VP.

Oracle Label Security mediates access to rows in database tables based on a label contained in the row, a label associated with each database session, and Oracle Label Security privileges assigned to the session. It provides access mediation on an application table after a user has been granted the standard Oracle database system and object privileges. For example, suppose a user has SELECT privilege on the fact table. If the user executes a SELECT statement on the table, Oracle Label Security will evaluate the selected rows and determine if the user can access them based on the privileges and access labels assigned to the user. Oracle Label Security also performs such security checks on UPDATE, DELETE, and INSERT statements. Labels can be applied to fact tables as well as to materialized views, where the materialized views increase performance and labels increase security, thus ensuring the flexibility, speed and scalability desired in data warehouse environments.

Label Benefits for the Data Warehouse

Oracle Label Security enables people who build and manage data warehouses to consolidate information from multiple sources into one, very large system, with the convenience and manageability and security of centralized administration. Because this security option to Oracle Database 10g is an application on its own, there is no need to do any PL/SQL programming. Oracle Label Security includes a graphical user interface, Oracle Policy Manager, to create and manage policies, labels and user authorizations. It enables consolidation, minimizes risk by enforcing security on the data itself, and provides granular security by controlling access to data down to the row level.

ADDITIONAL PROTECTION WITH ENCRYPTION

Data traveling over a network is not protected by the multiple layers of security that safeguard it inside the RDBMS. To this end, Oracle supports encryption of network traffic, as it has since Oracle7. Industry-standard algorithms, such as DES, Triple-DES and AES, protect Oracle network traffic, both between clients and servers and between database servers.

Among other database security measures, Oracle also protects data stored in warehouses via encryption within the database. Although encryption should never be used as a substitute for effective access control, one can obtain an additional measure of security by selectively encrypting sensitive information, such as credit card numbers, before it is stored in the database. The Oracle Database provides a PL/SQL package that encrypts and decrypts stored data. The `dbms_obfuscation_toolkit` package supports data encryption using industry-standard algorithms:

- **DES encryption:** The toolkit includes procedures to encrypt and decrypt using the Data Encryption Standard (DES) algorithm.
- **Triple-DES encryption:** The toolkit supports Triple DES (3DES) in two- and three-key modes.
- **MD5 cryptographic hash:** This one-way cryptographic hash algorithm ensures data integrity.

It also provides a random number generator, because random numbers are required to generate secure encryption keys.

Encrypting stored data can provide the assurance of securing data regardless of access method—that is, even if a person accesses the operating system files where other mechanisms for database-enforced security cannot protect it. However, encrypting data inside a data warehouse can be complex, and it usually adds overhead to the system. Additionally, the encryption keys must be stored somewhere—in an application, in a file, or in a table—and managing keys is widely recognized as a very difficult security problem. Many risks that appear to be managed with stored data encryption are better solved with the combination of

other proven security solutions, such as strong user authentication, network encryption, granular access controls, and auditing.

USER ACCOUNTABILITY

Data warehouses hold massive amounts of financial information, company secrets, medical diagnoses, credit card numbers, and other personal information. Because the data warehouse is a hotbed of critical information, it makes a lucrative target for legitimate users who need data access to do their jobs and for malicious users who desire access to its valuable data. Developers and administrators who build and manage warehouses need to maintain a record of system activity, both to be able to roll back where an error has occurred and to ensure that users are held accountable for their actions. Auditing selected sensitive information and user actions helps to keep users accountable and data protected. Further, auditing helps deter unauthorized user behavior that may not otherwise be prevented.

Auditing the Data Warehouse

The standard audit facility in Oracle allows the auditing of database activity by statement, by use of system privilege, by object, or by user. For example, one can audit activity as general as all user connections to the database, and as specific as a user updating a table. One can also audit only successful operations, or unsuccessful operations. Auditing unsuccessful select statements may catch users testing their access boundaries or snooping for data they are not privileged to see.

Performance cannot be sacrificed in a data warehouse, and, as such, SQL statements are parsed once for both execution and auditing, not separately. The granularity and scope of audit options allow you to record and monitor specific database activity without incurring the performance overhead that more general auditing entails. And, by setting just the options of interest, you can avoid catch-all audit methods which intercept and log all statements, and then filter them to retrieve the ones of interest. Because queries against a data warehouse generally take longer to process than queries in OLTP systems, any performance impact of auditing is negligible. On the other hand, warehouses contain massive amounts of data, so it is crucial to narrow the scope of auditing to the most important data.

Fine-grained Auditing for Data Warehouses

Oracle expands upon the standard auditing capabilities of the database with extensible Fine-grained Auditing (FGA). The fine-grained audit feature enables organizations to define specific audit *policies* that can alert administrators to misuse of legitimate data access rights. Standard auditing mechanisms audit at session, privilege or object-level, where fine-grained auditing hones in on the particular data of interest. Fine-grained auditing policies allow you to specify the data access *conditions* that trigger the audit event, and use a flexible event handler to notify administrators that this event has occurred.

For example, a warehouse of investment banking data may allow the bank's financial analysts to access pre-IPO customer financial records, but audits access when assets greater than \$100M are selected, inserted, updated or deleted. The audit policy ("where ASSETS > 100000000") is applied to the CUSTOMERS table through an audit policy interface (a PL/SQL package called DBMS_FGA).

These policies can identify a relevant column for auditing—an "audit column". An audit column helps reduce the instances of false or unnecessary audit records, because the audit need only be triggered when a particular column is referenced in the query. The 10g release allows you to define one or more relevant columns in an FGA policy. Support for relevant columns is particularly important where personal information is stored. For example, an organization may only wish to audit access to bank account number when the account owner's name is also accessed, because accessing bank account number alone may not be a privacy violation without the person's name, the bank, or other corresponding data.

The Oracle Database 10g release captures the exact SQL text of the statement the user executed. This is true for both "standard" (when the AUDIT_TRAIL parameter is set to DB_EXTENDED) and fine-grained audit trails (by default).

Finally, fine-grained audit policies have an integrated event handler, enabling you to determine how to handle a triggering audit event. An audit event could be written into a special table for further analysis, or could activate a pager for the security administrator. The event handler allows organizations to fine-tune their audit response to appropriate levels of escalation. Fine-grained auditing helps identify the "needle" of data (in the haystack of the data warehouse) that requires a record of who saw or modified it.

Benefits of Auditing the Data Warehouse

It would be unwise to audit too broadly across a warehouse, for it would result significantly more audit records than administrators could hope to read or understand. Fine-grained auditing enables data warehouse administrator to hone their auditing to capture and identify particular, specific data access of concern. Because you can base your auditing on any condition that can be expressed in a function or where clause, auditing occurs less often but more accurately than ordinary audit mechanisms. With its integrated event handler, it can immediately alert security personnel where an intrusion has been detected. Auditing is the last line of defense in a layered approach to security, and the Oracle database leads the industry with the granularity and customizability available in a data warehouse.

CUSTOMER EXAMPLES

Case Study #1: Data Warehouse for a Government

A regional government created a data warehouse to analyze key metrics such as budgetary and expenditure information. Since each department should only be able to view the financial information for their own department, the data warehouse

team implemented Virtual Private Database. Using this approach, each of the 28 government departments has its own 'virtual data mart'; from the perspective of the end-users within a government department, the system appears to be a data mart containing only their departments' data. Additionally, the government's internal auditing department can use this same data warehouse to retrieve consolidated financial information across all departments. By using VPD, this government has provided a data-mart look and feel to each of its 28 departments, while benefiting from the cost-savings of implementing a single system (rather than 28 different systems) and gaining the ability to do consolidated analysis.

Case Study #2: A Data Syndicator

Data syndicators are companies whose business is selling data. Data syndicators exist in a wide variety of industries, and typically sell data such as customer demographic information or sales data. Data syndicators typically gather data from many sources and generally cleanse and enrich the data before selling it to their customers. The technical processes involved in data syndication are thus in many respects similar to those in a data warehouse.

A data syndicator using an Oracle data warehouse delivers marketing research information to its customers via the web. The security challenges of a data syndicator are two fold. First and foremost, the company provides each of its clients with data about that client's customers and products; one client must not be able to see another client's data. For the data syndicator, this is particularly important since it is likely that data syndicators clients are direct competitors. Second, within each client, the data syndicator provides different levels of access to the data (so that, for example, some end-users are limited to seeing data for a specific set of products). This security model is implemented using Virtual Private Database.

As in the previous case study, a major benefit of virtual private database is that it is a key enabler of consolidation. This data syndicator is able to serve 30 clients from a single data warehouse, rather than build 30 separate data marts. Moreover, the data syndicator can very quickly enhance their solution. Adding a new report, for example, to all 30 clients is very simple, since the report only needs to be developed and deployed on one system.

Case study #3: A major financial institution

A major financial institution has built a data warehouse focused on customer profitability. This system contains information for over 2 million customers, gathers data from over 300 sources, and is utilized by 20,000 bank employees worldwide.

Virtual Private Database enables this financial institution to maintain a complete separation of its customers' data, so that a given employee is only able to view information about a specific set of customers. This VPD-enabled data warehouse ensures that the customer data remains secure, without imposing any significant performance penalties. Moreover, because virtual private database operates inside

the database, the financial institution has the flexibility of using any tool against the data warehouse while maintaining the security of the information.

SUMMARY

A data warehouse depends on making the best, most up-to-date information available to users when they need it. It must provide customers and employees with access to information, in a way that is fast, scalable and secure. Oracle Database 10g, with key features such as Virtual Private Database, provides seamless, consistent security for enterprise data warehouses.



Security and the Data Warehouse

April 2005

Author: Kristy Browder Edwards, George Lumpkin

Contributing Authors: Mary Ann Davidson, Paul Needham, John Heimann

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

oracle.com

Copyright © 2005, Oracle. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice.

This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle, JD Edwards, and PeopleSoft are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.