# Oracle9*i*™ Replication

*An Oracle White Paper*
*June 2001*

ORACLE

## THE NEED FOR REPLICATION

Increasingly, companies are turning to Oracle replication to provide the high availability, scalability, and improved performance required by their applications. The rapid growth of the Internet further increases the need for improved replication technology. From distributed website integration to mass deployment over the Internet, Oracle replication provides the technology required to meet these needs.

## PRODUCT SUMMARY

Replication is the process of copying and maintaining database objects in multiple databases that make up a distributed database system. Changes applied at one site are captured and stored locally before being forwarded and applied at each of the remote locations. Replication provides users with fast, local access to shared data, and protects the availability of applications because alternate data access options exist. Even if one site becomes unavailable users can continue to query or even update the remaining locations.

Oracle replication is designed to support a wide variety of applications, often with very different requirements. For example, sales force automation, field service, retail and other mass deployment applications typically require data to be periodically synchronized between central systems and very large numbers of small, remote sites often operating in a disconnected manner. While at the other end of the spectrum, applications such as call centers and Internet systems require data on multiple servers to be synchronized in a continuous, near-real time manner to ensure that the service provided is up and available at all times. Oracle provides a sophisticated conflict detection mechanism and a comprehensive set of automated conflict resolution routines to ensure data convergence throughout the replicated environment.

Oracle replication enables both mass deployment and server-to-server configurations to be integrated into one coherent environment so that, for example, salesforce automation applications and customer service call centers can share data. Additionally, Oracle® Enterprise Manager provides an easy to use graphical user interface, which provides users with a single point of control for their entire replicated environment.

## SERVER-TO-SERVER SYNCHRONIZATION

For server-to-server configurations supporting multiple web sites, call centers and other applications, Oracle replication enables data, such as user account information or customer service records, to be moved quickly and in large volume between locations.

To keep these locations as up to date as possible, Oracle replication supports near real-time data propagation between n-way connected sites, or masters. This form of multiple master site, or multimaster, replication supports full table, peer-to-peer replication between master tables. Oracle replication supports standard Oracle data types, such as VARCHAR2 and NUMBER. Oracle9*i* replication extends this support to include most user-defined data types, such as object tables and tables with nested tables.

Master tables at all sites can be updated. Changes applied to any master table are propagated and applied directly to all other master tables. With asynchronous multimaster replication, each change made by a user is captured and stored locally, and then later forwarded and applied at each remote location. Multimaster replication uses deferred remote procedure calls (RPCs) as the underlying transport mechanism to propagate and apply changes.

Changes to multiple master tables are applied in a transactionally consistent manner to ensure data and referential consistency. Changes are propagated either immediately in an event-based manner, or at specified points in time when connectivity is available, or when communications costs are lowest, e.g., during evening hours. Failures of any one master site will not block propagation of changes between other master sites. If a remote system is unavailable, the deferred RPCs propagating changes to that system remain in their local queue for later execution.

If two sites update the same data element within the same replication interval, an update conflict will occur. To ensure convergence, Oracle detects and resolves update conflicts, so that the data element has the same value at every site. Users may choose one of Oracle's many built-in conflict resolution routines, such as "latest timestamp" or "site-priority", to resolve potential conflicts. Users can select different methods for different tables, or even different groups of columns within a table. Users can also create their own routines to employ resolution rules tailored to their particular business needs.

Oracle9*i* replication has been carefully designed to provide the optimal performance required by users of multimaster replication, while still ensuring data integrity throughout the replicated environment.There are three key components in maximizing performance to provide near real-time propagation:

- Efficiently capturing and applying replicated changes
- Maximizing propagation performance through parallelism
- Minimizing the amount of data that must be sent when propagating changes

## Capturing and Applying Replicated Changes

When capturing and applying replicated changes, it is important not to slow down existing applications by introducing unnecessary overhead on the existing system. Additionally, as a replicated system grows, users require a solution that will continue to scale, and that will be easy to manage and maintain.

Replication uses triggers to capture replicated transactions and store them in a local queue. Once these changes have been propagated to the remote location, packages are used to apply the replicated changes at the target. To improve performance and reduce overhead, these triggers and apply packages have been internalized. Because they are written in C and compiled directly into the Oracle kernel, these lightweight internal mechanisms create virtually no overhead on the replicated table and scale readily as your system grows. Oracle's change capture and apply mechanisms are completely integrated with the Oracle Server, making management easy. There are no special, external components to configure and maintain.

.

## Propagating Changes

After capturing changes to replicated data, the changes must be propagated to each remote location to be applied. Oracle replication uses multiple sessions to parallelize the transaction stream to each remote location. Oracle replication automatically parallelizes the propagation stream to each location, while still preserving transactional integrity, by determining the dependencies between the transactions.

.

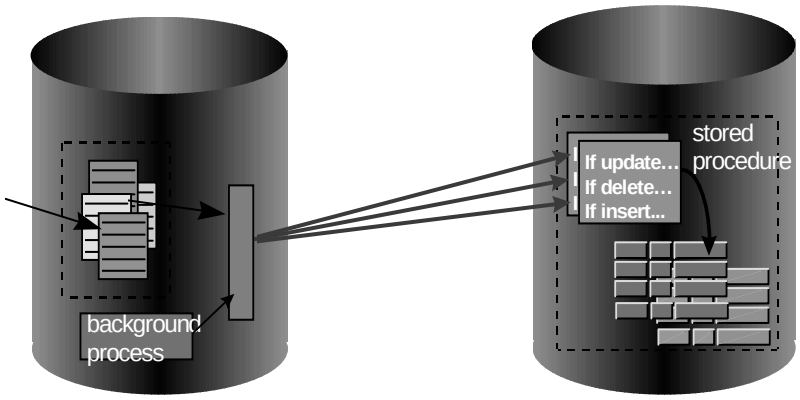



*Figure 1. Parallel Propagation Greatly Improves Replication Performance*

When propagating replicated data, it is important to preserve ordering between dependent transactions. The easiest method of preserving ordering between transactions is to apply the transactions at each remote location in the same order that they were applied locally, by serializing the transaction stream. This method,

however, sets a low threshold for the transaction volume that can be accommodated by a replicated environment.

Oracle uses a lightweight coordinator process at the local site to determine the dependencies between transactions before they are propagated. Parallel propagation makes use of the pool of available parallel slave processes, in a manner similar to parallel query. The coordinator tracks the transaction dependencies, allocates work to these server processes, and tracks the progress of the transactions. A transaction is not propagated until after the transactions that it depends on have been propagated. Multiple transactions that have no dependencies can be propagated (and committed) in parallel—reducing bottlenecks. Oracle9*i* further improves the performance and scalability of parallel propagation by tracking dependencies at the row level. This allows the coordinator to track dependencies and order changes more efficiently when applying the deferred transaction queue.

### Minimizing Data Propagation

The less data that you need to propagate, the faster you can propagate it. This is especially true for tables with long rows or columns containing large objects, such as binary images. With Oracle replication users can choose to send only the minimum amount of data necessary to successfully apply a change at a remote location.

The minimum set of values that must be sent to apply a change includes the following:

- column values necessary to uniquely identify the row being updated—typically the primary key columns
- the new values for any changed columns
- the column values needed by conflict resolution routines—such as, a timestamp

For example, suppose a user has the following table specification, and that she has chosen to use the Oracle-supplied latest timestamp conflict resolution method to always apply the most recent change in the event of conflicting updates to the same row.

```
CREATE TABLE employee
      (empno      NUMBER      CONSTRAINT pk_emp PRIMARY
KEY,
       ename      VARCHAR2(10)      CONSTRAINT nn_ename
NOT NULL
                  CONSTRAINT upper_ename
                    CHECK (ename = UPPER(ename)),
       job  VARCHAR2(9),
       mgr  NUMBER      CONSTRAINT fk_mgr
                    REFERENCES scott.emp(empno),
       hiredate   DATE  DEFAULT SYSDATE,
       picture    BLOB,
       timestamp  DATE)
```

In this example, Oracle would only need to send the values of the primary key (EMPNO) and TIMESTAMP columns, as well as the new values for any columns that were changed as the result of a transaction.

## MASS DEPLOYMENT APPLICATIONS

Another rapidly growing market for replication technology is mass deployment. As more and more companies recognize the need to automate their front-line operations, Oracle continues to add sophisticated new functionality designed to meet this need. There are four key challenges to supporting these types of mass deployment applications:

- The remote sites must be able resync on demand, as quickly as possible, whenever connectivity is available.
- There must be support for hierarchical configurations.
- Each user must only receive the particular subset of data that he or she needs.
- It must be possible to centrally deploy and administer hundreds or even thousands of these remote sites.

Oracle updatable materialized views, or snapshots, are designed to meet these requirements. Materialized views can be defined to contain a full copy of a master table or a defined subset of the rows in the master table that satisfy a value-based selection criterion. With Oracle9*i*, you can even create materialized views of tables defined using user-defined data types.

Materialized views are refreshed from the master at time-based intervals or on demand. Updates to materialized views are propagated and applied to materialized view masters using deferred RPCs as the underlying mechanism. Any changes to the master table since the last refresh are then propagated and applied to the materialized view. Multiple materialized views are refreshed in a transactionally consistent manner to ensure data and referential integrity.

### Support for Disconnected Operations

Decreased connection times are a key requirement in most disconnected operations. Mobile users are typically running on a small footprint database with only minimal access, such as a hotel phone line, to be able to resynch with headquarters.

Oracle recognizes that mobile users need to be able to resync their databases as quickly as possible, whenever a network connection is available. Thus, materialized view refresh has been optimized to perform as quickly as possible. The materialized view refresh optimizations are especially targeted at applications that use substantial numbers of tables, such as, one hundred to several hundred tables, high latency WAN or Internet network connections, and complex rules defining

the data subsets to be maintained involving subqueries accessing tables in multi-level child / parent relationships.

Performance improvements center around two types of optimizations. First, are low level protocol optimizations that minimize the number of message exchanges required to replicate changes. And second, are optimizations that enable the materialized view refresh mechanism to very efficiently detect when no changes have been applied to certain tables, such as, lookup tables that are seldom modified, and hence nothing needs to be replicated.

### Multitier Materialized Views

With Oracle9*i* you can create an updatable materialized view based on another updatable materialized view. This multitier capability is useful where network connectivity between sites is an issue. For example, in a Headquarters, Regional Office, Branch Office configuration, individual Branch offices may not have network connectivity to the Headquarters site. In a multitier updatable materialized view configuration, each middle tier updatable materialized view performs as a master site for any subsequent updatable materialized views. In particular, the middle tier materialized view performs conflict resolution for any updates that come from the subsequent materialized views. At each site that is acting as a master site, conflict detection and resolution occurs. If a change that originated from the lower level site (a Branch office in the earlier example) is rejected, the middle-tier site data values are refreshed to the lower-level site, via the normal refresh mechanism. Changes percolate up to the top-level sites by successive refreshes of the intermediate level sites.

### Defining Unique Subsets with Row- and Column-Level Subsetting

To further reduce connection times and minimize the footprint at the remote database site, it is critical that each remote user only receive the subset of data that he or she requires. With updatable materialized views users can define not only which tables will be replicated to a given location, but which columns and rows of those tables will be replicated. Oracle9*i* updatable materialized views provide a solution that lets users:

- Easily define and maintain the subset of the database needed by each remote site
- Ensure that each user is receiving only the information that he or she wants and needs
- Allow remote users to send and receive incremental changes to/from their main office
- Support many-to-many relationships (for example, multiple salespeople working on the same accounts and multiple accounts per salesperson)

- Easily change these relationships (for example, if a customer moves to a new sales territory)
- Make major changes in the field organization, without having to rebuild and redeploy the application

When users create materialized views they must somehow define the subset of data that they want to replicate to each site. With Oracle replication, users define the subset using a SELECT statement as part of a CREATE MATERIALIZED VIEW command. The columns listed in this SELECT statement determine which columns are replicated to a given materialized view site. For example, if an inventory table contains descriptive information about each item, as well as an image of that item, users might choose to save space at the remote site by not replicating the image column.

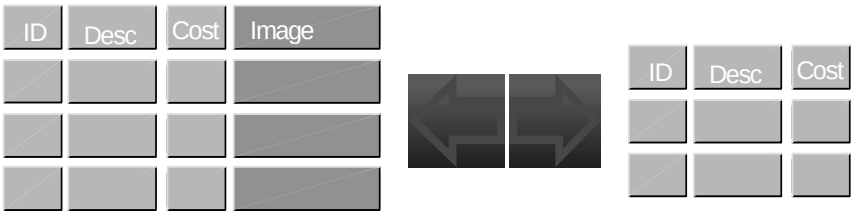



*Figure 2: Oracle9i Replication Supports both Column- and Row-level Subsetting*

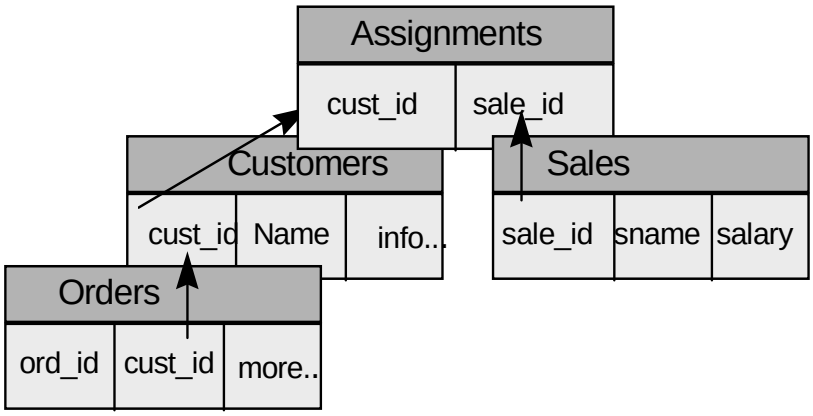
To further restrict the amount of data that is replicated, users might choose to include a subquery as part of the SELECT clause. By using subquery subsetting, users can easily define and maintain unique subsets of data for large numbers of remote sites. Users can choose to replicate only those rows that satisfy their unique requirements, for example, a user might choose to replicate only those orders for customers who are assigned to her.

Subquery subsetting allows users to define their assignment information in one location, and reference it from the other tables using subqueries. Subqueries "walk up" the foreign key references already maintained on the tables, to allow a "child table" to reference assignment information maintained by a "parent table." Support for many-to-many relationships between tables, as well as AND conditions across multiple rows and OR conditions within rows, yields maximum flexibility and protection against change.

Users don't have to sacrifice performance to take advantage of subquery subsetting. Subquery materialized views, including those with many-to-many relationships, can be refreshed using Oracle's fast refresh mechanism, which allows materialized views to both send and receive incremental updates from their associated master tables.

As shown in the following example, if a user wanted to create a materialized view of the "orders" table, that contained only the orders for customers assigned to salesmen with names beginning with "jones" and salary > $1000, his materialized

view definition would look similar to the one shown below:




```
CREATE MATERIALIZED VIEW orders FOR REFRESH FAST AS
  SELECT * FROM orders@master.db o WHERE
EXISTS
  (SELECT cust_id FROM customers@master.db c
   WHERE o.cust_id = c.cust_id AND
EXISTS
  (SELECT cust_id FROM assignments@master.db a
   WHERE c.cust_id = a.cust_id AND
EXISTS
  (SELECT * from sales@master.db s
    WHERE sname like 'jones%' and a.sale_id =
s.sale.id)));
```

*Figure 3: Subquery Materialized view Let You Easily Define Unique Subsets*

Note that using a separate table to assign customers to salespeople insulates users from having to change their materialized view definitions if changes are made to the sales organization. For example, if customers were assigned by geographic region but will now be assigned by sales volume, a user would simply update the assignments table to implement this change. These changes would be reflected in the field the next time a salesperson performed a fast refresh. There would be no need to modify the schema, to edit the materialized view definitions, or to recreate the materialized views.

### Centralized Administration of Remote Materialized view Sites

Administering hundreds of disconnected users is probably the greatest challenge facing companies implementing mass deployment applications. Oracle replication meets this challenge by providing deployment templates and a simple off-line instantiation mechanism that allows users to define once and deploy to hundreds or even thousands of sites. Additionally, users can take advantage of the replication node in Oracle Enterprise Manager to monitor and maintain these disconnected sites.

Materialized view deployment templates greatly simplify the task of deploying and maintaining hundreds of remote materialized view sites. Deployment templates allow users to define a collection of parameterized materialized view definitions at

a master site. When a remote user connects to a master site, he sees a list of templates to which he has access. For example, there might be one template available for the sales force and another for the field service representatives. The user can then choose to instantiate the template, at which time the appropriate materialized views will be created and populated at the remote site. The appropriate parameter values (for example, sales territory or customer support level) can either be supplied by the end user or by a look-up table maintained at the master site.

To decrease connection times, the remote user may choose to have the materialized views be "pre-instantiated" at the master site. The materialized view user would then simply transfer these existing materialized views over to her remote location, disconnect, and begin working. The next time she performed a fast refresh she would see any changes that had been made at the master site since the materialized view was instantiated.

## HYBRID APPLICATIONS

Multiple master replication and updatable materialized views can be combined in hybrid configurations to meet different needs. Specifically, materialized view masters can be replicated in multiple master configurations. This allows full-table and table subset replication to be combined in one system.

For example, multiple masters can replicate between two hub sites supporting two geographic regions. Read-only or updatable materialized views can be defined on the masters to replicate full tables or table subsets to sites within each region. These regional sites could then act as masters for materialized views needed by disconnected users in the field, such as sales personnel. Users can easily add master or materialized view sites as needed to meet the scalability needs of their rapidly growing environments. Additionally, this type of configuration allows the master sites to function as fail-over sites for each other.

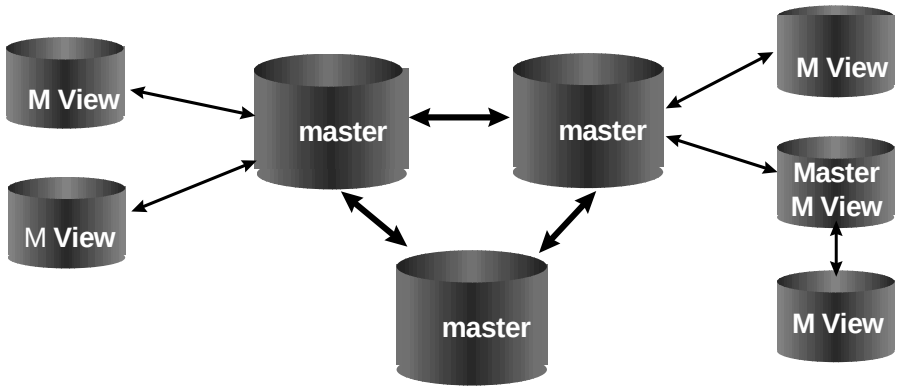



*Figure 4. Oracle's Unique Multi-hub and Spoke Architecture Supports the Widest Variety of Applications*

## ALTERNATIVE REPLICATION MECHANISMS

In addition to asynchronous multi-master and materialized view site replication, Oracle provides two alternative mechanisms for propagating data-level changes between sites:

- procedural replication
- synchronous replication

*Procedural Replication*—Procedural replication also uses deferred remote procedure calls. Occasionally it may be necessary to perform large numbers of updates in a serial, batch-oriented manner to replicated data. For example, perhaps once a quarter or once a year it may be necessary to purge old order data that may no longer need to be kept online. Replicating each individual row change across multiple sites could be very inefficient. Instead, by using procedural replication, users can call a procedure at one location, and ensure that an identical procedure will be executed at each replicate site to perform the updates directly.

*Synchronous Replication*—When users make a change to a replicated table using synchronous replication, Oracle ensures that the change is successfully applied at both the local table and at any replicated copies of the table, or the transaction is rolled back. Synchronous replication is most useful in situations where users have a stable network and require that their replicated sites remain continuously synchronized.

## MAINTAINING A REPLICATED ENVIRONMENT

Within the database, Oracle9*i* replication's distributed schema management facility provides the mechanisms for defining and changing a replicated environment. The replication node of Oracle Enterprise Manager provides a graphical user interface for this facility. The distributed schema management facility stores the definition of a replicated environment in the replication catalog, which can also be queried using standard SQL commands from a variety of tools.

### Replication Catalog

The replication catalog provides a single, consolidated repository containing the metadata that defines the replicated environment. The replication catalog is itself replicated to multiple sites to ensure high availability and easy local access by authorized users wherever they may be located.

The replication catalog defines the database objects being replicated, the sites where they are replicated, and the mechanisms used to support their replication. Database objects include the tables that contain replicated data and the object definitions for other supporting components including indexes, views, procedures, triggers, and synonyms. An extension to Oracle's standard data dictionary, this metadata is open and accessible. It can easily be queried using standard SQL.

## Distributed Schema Management

Oracle9*i* replication's distributed schema management capability allows replicated environments to be defined and changed from a single control point. It automatically replicates and applies data definition language (DDL) commands to multiple sites. It also automatically activates the internal triggers and internal procedures used to support replication.

Distributed schema management allows users to perform operations such as:

- Add an index or check constraint to a replicated table everywhere
- Add all the database objects in a replication group to a new site
- Alter a procedure or view definition everywhere

Distributed schema management operations are performed and controlled from one site called the master definition site. It automatically pushes DDL to all master sites and allows materialized view sites to pull down and apply DDL on demand. Because it uses and maintains the replication catalog, which is itself replicated to multiple sites, the master definition site can be changed in case of failure. Distributed schema management operations are initiated by GUI-based management tools from Oracle or other vendors through an open procedural API.

## Replication Management Tool

The Oracle Enterprise Manager console contains a replication node that lets a replication administrator graphically configure, schedule, and administer a replicated environment from a single location. This point-and-click interface lets users define groups of database objects that need to be replicated and managed as a unit. These objects can include not just tables, but their supporting objects as well. After creating a group, users can drag and drop it onto other databases to add new replication sites to their environment. If a user adds or removes objects from a replication group, the changes are automatically deployed at every site.

Oracle Enterprise Manager also helps users troubleshoot and resolve error conditions. A replication administrator can view the deferred transaction queue at each location, and reschedule or force immediate execution of these transactions as needed. The administrator can also view outstanding administrative requests for each location. Oracle9*i* provides improved performance monitoring and enhanced reporting for replication users.

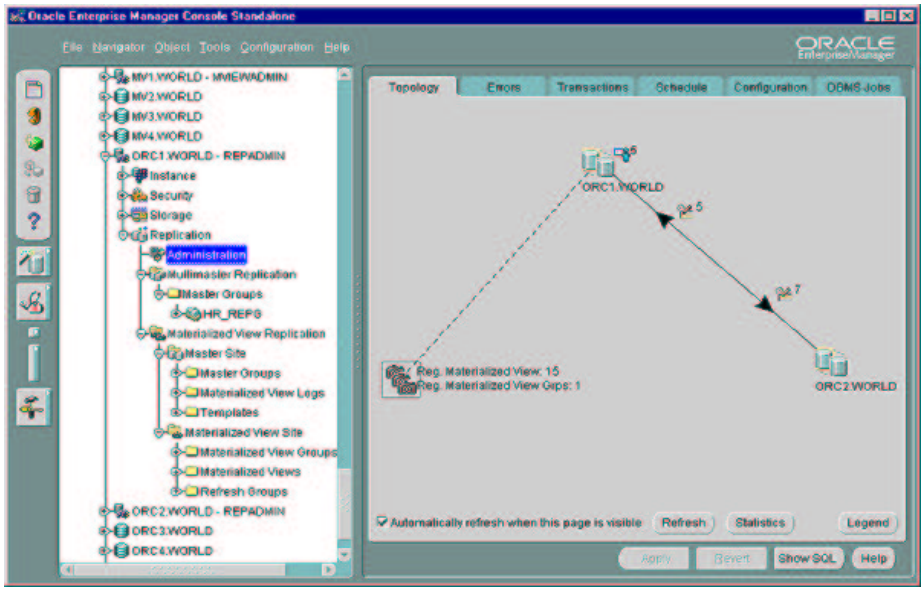



*Figure 5. Oracle Enterprise Manager Helps Users Configure and Maintain a Replicated Environment*

## SUMMARY

Oracle9*i* replication delivers the industry's most comprehensive replication capabilities. All replication mechanisms were designed from the beginning to be bi-directional. Replication change capture and apply functions run directly within the Oracle Server executable itself (in C code) for maximum performance and minimal resource utilization. Robust, comprehensive update conflict detection and resolution mechanisms protect the entire replicated environment. And a single tool - Oracle Enterprise Manager - manages it all.

From mass deployment applications, such as sales force automation, to applications requiring data synchronization between servers supporting, for example, multiple websites or call centers, Oracle9*i* replication supports them all in one integrated environment.

.

**Oracle9*i* Replication**
**June 2001**
**Author:Maria Pratt**
**Contributing Authors: Patricia McElroy**

**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**

**Oracle Corporation provides the software that powers the internet.**