White Paper

ActiveKnowledge™ - Innovative Performance and Scalability Solution for the Enterprise

Toll Free: 866 - 300 - 4022
Email: sales@active-base.com
Visit us: www.active-base.com

Table of Contents

Databases growth trends

Over the past few years, organizations have made enormous investments in applications designed to store every piece of information in a database. They have deployed various applications managing every aspect of business operations, created terabyte-scale data warehouses, and spent millions on multi-processing servers powerful enough to process the resulting volumes.

Even the most powerful database servers cannot always keep pace with the demands imposed by scores of financial analysts, marketing researchers, strategic planners and customer relationship managers.

As a result, many organizations are literally choking on the terabytes of data and processing loads that drive resource usage beyond acceptable limits, and exceed system capacities. System performance slows to a crawl, and the lights on the DBA's phone begin to flash as the complaints roll in.

These increased application performance and high-availability demands, combined with the freedom of users to issue ad-hock queries unaware of their complexity, are encountered within IT departments suffering from tight budgets as well as insufficient manpower to maintain high performance levels.

Confronted with this challenge, IT managers are looking for solutions that improve application performance and high-availability in an 'out-of-the-box' approach – that incurs no additional work load on the existing IT workforce, and are immediately installed with no hidden modifications or maintenance costs.

They seek solutions that have a profound impact on performance, stability and high-availability, and that increase efficiencies of past infrastructure investments - an immediate ROI with no risks involved.

Existing Strategies for Performance Improvement

Database administrators enjoy a few alternatives when it comes to improving processing performance.

The hardware upgrade (big hammer approach) strategy of upgrading the database server hardware was the ultimate solution when budget was not an issue. This approach demands large capital allocations. Adding a CPU is not expensive; but the associated software licenses make it a costly solution. This solution requires enough budget for tedious planning, analysis, installation, testing and with long ROI. Organizations try to use other performance improvement solutions in order to benefit from their existing infrastructure for a longer period of time.

The Alternate Database approach (replication) is an approach that copies parts of the database to a replication. These replicated

databases are used by Business Intelligence tools that do not require real-time and up-to-date information. Replications require purchasing additional hardware and software, and the management and overhead of data synchronization between the different copies of replications. In addition, replication requires the management of access to the different databases by different users running different applications. This solution is expensive, complex and requires on-going maintenance and tuning.

Tuning Database approach (Performance Management Tools)

Using database and applications tuning tools is a solution that many enterprises deploy. They use monitoring tools that help them changing database general parameters - like size of general storage area, size of local pools and indexes that are recommended to be added to specific tables – all of which provide timely and limited performance relief.

On the other hand, monitoring tools identify a small set of application requests that cause most of the overhead on the database server and the longest applications response time. They even suggest corrections for these statements in a way that the DBMS will be executed more efficiently while preserving the SQL results **– YET THE DBA IS UNABLE TO CORRECT THE PROBLEMATIC REQUESTS – EITHER BECAUSE THE APPLICATION IS A CLOSED PACKAGE, THE PROGRAMMERS ARE NOT AVAILABLE OR BECAUSE THE REQUESTS ARE REPORTS AND AD-HOC QUERIES CREATED AND RUN BY AN END USER IN BUSINESS INTELLIGENCE ENVIRONMENTS.**

ActiveKnowledge™

ActiveKnowledge simple and easy-to-deploy software speeds problematic reports and ad-hoc queries on their-way to the databases, tuning them and applying performance optimization policies in real-time.

Installed transparently between applications and the databases, it boosts performance by identifying in real-time problematic SQL requests on-their-way to the databases, analyzing and correcting them automatically - without rewriting the applications or changing the databases.

Offensive or unauthorized requests are blocked before reaching the databases, returning customized notification to the user. Blocking these statements prevents performance degradation and eliminates security breaches.

In addition, ActiveKnowledge provides detailed usage patterns and audit-trails of users and requests.

ActiveKnowledge is installed and configured in less than an hour, without any modifications imposed on clients\applications or databases. The installation into existing database infrastructure is simple, quick, and easy, and does not require any installation downtime or application customization.

ActiveKnowledge Results

- Improved response time of queries, reports and requests by up to 500% using an "out-of-the-box" approach. The profound impact translates directly into significant improvements in business performance, with no added demands on end users.
- Increased database capacity, support more data/users from existing HW/SW resources.
- Improved visibility to users, applications and database objects usage patterns using detailed audit trail and applications activities reducing licensing and security.
- Performance gains is easily achieved by DBA's and other non-expert users including report writers, developers and super users.
- Reduced expertise is required for improving performance

How does ActiveKnowledge accomplishes these results?

ActiveKnowledge operation automates and streamlines the following actions:

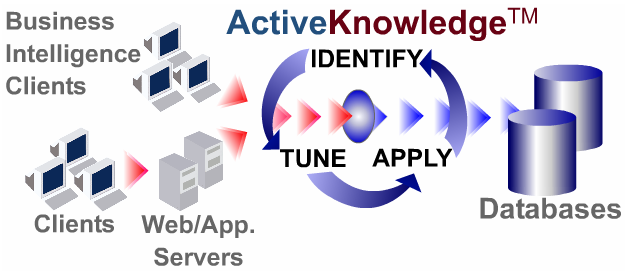
1. Identify - ActiveKnowledge automatically identifies long running queries, problematic requests and reports. It provides a full audit trail on all SQL statements and database execution plan operations used.

2. Tune - ActiveBase SQL expert tunes problematic statements by adding database 'Hints', identifying best execution alternative.

3. Apply - ActiveKnowledge automatically intercepts problematic SQL statements in real-time, and corrects them on-their-way to the databases, using best execution alternative found (using ActiveBase Expert or 3rd party tools). ActiveKnowledge applies SQL optimization policies without intrusively 're-coding' the applications or manually rewriting ad-hoc queries and business reports.

ActiveKnowledge can be installed on the database server or on a server near it. Each server can manage up to tens of different applications and databases.

It supports all applications running against an Oracle database version 7.3 or higher, on all major platforms. ActiveKnowledge can be installed on Windows, Linux, Solaris and HP-UX server.



Topology

ActiveKnowledge is built on Database Network Router™ software, transparently installed between the applications and the databases. It acts as a router - all communications between applications and databases are routed through it, as it works on the database network protocol layer. ActiveKnowledge is completely transparent - clients and application servers see

ActiveKnowledge as their database server, The database servers see ActiveKnowledge as any other application/client.

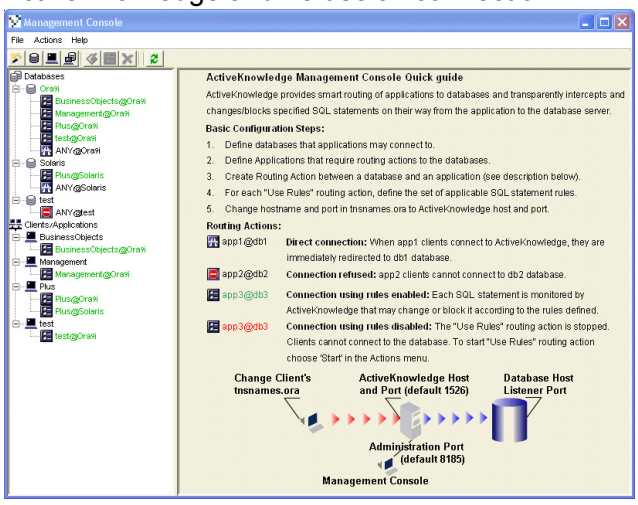
Remote management console enables easy remote configuration and management.

Functional Layers

ActiveKnowledge is built of three independent functional layers

Application Connection Switching Layer

All clients/applications connect to the databases using a single ActiveKnowledge listener port (called ActiveKnowledge Port, default port number: 1526). Using client connection information, ActiveKnowledge identifies the database service name or SID and routes the connection accordingly to the database server. The Switching layer enables administrators to create different routing types and optimization policies customized for each application program name. Possible routing types include 'apply optimization rules', 'connect directly' - bypassing ActiveKnowledge and 'refuse all connection'.



Optimization Rules Layer

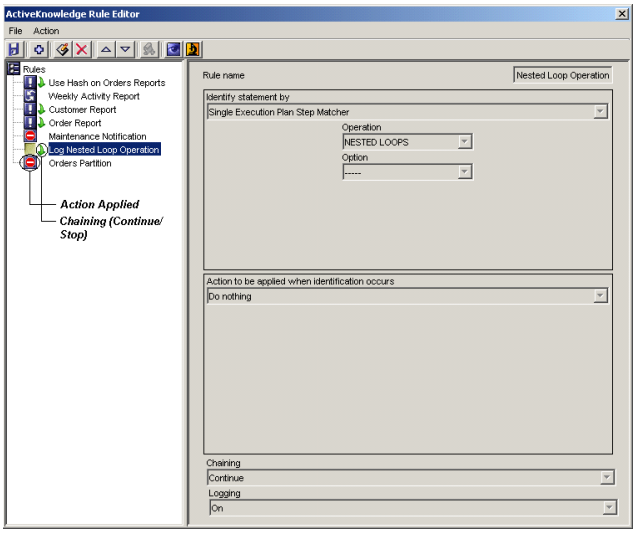
Enables to define and manage optimization rules and management policies, automatically applied on statements on-their-way to the databases. Possible rules include applying database 'Hints', statement rewrites and blocking offensive statements.

Each optimization rule is a two step process:

a. **SQL identification:** SQL identification methods include statement exact match, syntax match (includes SQL parsing), execution plan match (full plan match, partial or single execution step match), PL/SQL matcher, partition range matcher and Oracle cost matcher.

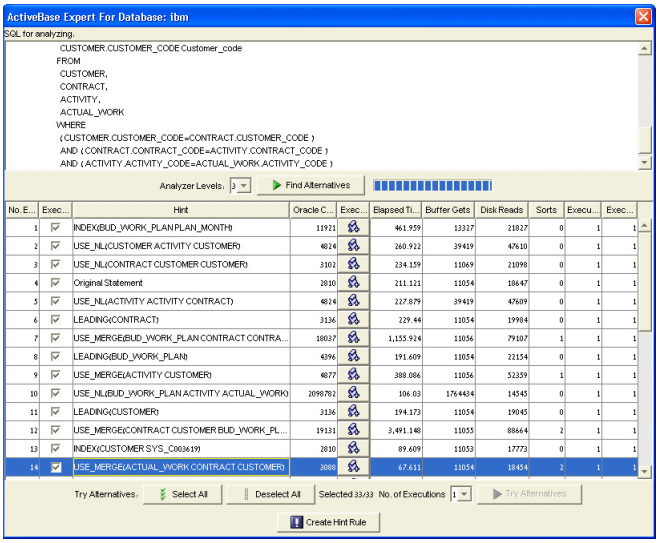
b. **Action applied on the SQL statement when identification occurs:** actions include adding a Hint, rewrite or block.

For defining optimization rules the ActiveBase SQL Expert can be used or applying tuning suggestions from other 3rd party tools.



ActiveBase SQL Expert Layer

The ActiveBase Expert tunes SQL statements by analyzing all possible hints that generate a unique execution plan in the database. These possible hints are presented as a list of optimization alternatives, sorted according to hint type and Oracle cost. ActiveBase Expert benchmarks the optimization alternatives to verify each alternative's execution time and CPU/IO load. By selecting the best alternative and clicking on the 'Create Hint Rule' button, an optimization rule is created in the Optimization Rules Layer, automatically applied on incoming statements – concluding the performance cycle.



Configuration

1. Clients/Applications are defined to use ActiveKnowledge server host name and ActiveKnowledge port as their primary database address with the database server as a fall-back using Oracle client's failover capability (to eliminate single point of failure).
2. Database names (Database addresses) and Clients/application names (application's program file names) are defined in ActiveKnowledge.
3. For each Application@Database a routing type can be defined.
4. Client connections are routed through ActiveKnowledge to the database servers.
5. Each statement is checked in real-time for compliance with one of the optimization rules. When such match occurs, the action defined in the rule is applied and the statement is optimized and forwarded for execution in the database, or blocked.

High Availability Solution

ActiveKnowledge eliminates the risk of failure. Even if ActiveKnowledge server has been completely removed from the network, an Oracle automatic client fail-safe mechanism is activated whereby all clients are immediately routed to the original database server.

ActiveBase can also be installed on a cluster, with a hot backup. The propagation delay added by ActiveKnowledge is less than one millisecond, making it very attractive for all applications.

activebase