



Orange Improves Performance with ActiveBase

“ActiveBase Performance solution delivered end-user productivity improvements and resulted in substantial Data warehouse resources savings, which enabled us to expand our Data warehouse using existing server resources.

It accelerated key business reporting by a factor of x10 and more, enabling us to run them much more often which dramatically improved our visibility into our business” Limor Malay, DW Division Manager, Orange (IL)

Background

Orange (IL) is a leading mobile GSM communications operator under the Orange™ brand, part of the Hutchison Whampoa Limited multinational conglomerate.

Orange implemented Business Objects 5.X, 6.X using client, web and Broadcast Agents on Sun Enterprise server and Oracle 9i and 10g database.

The Data warehouse users have been experiencing performance and scalability issues due to the massive usage growth.

ActiveBase Performance™ server has been quickly and transparently installed on the Sun Enterprise database server, with no agents and no changes in applications/databases. It automatically applies performance and management rules on incoming requests in real-time, dramatically improving performance, control and scalability.



The Results

ActiveBase Performance rules have resulted in the following performance and scalability improvements:

1. During the first month of operation, ActiveBase Performance have improved overall response time by 50%! - From 8.6 down to 4.3 minutes.

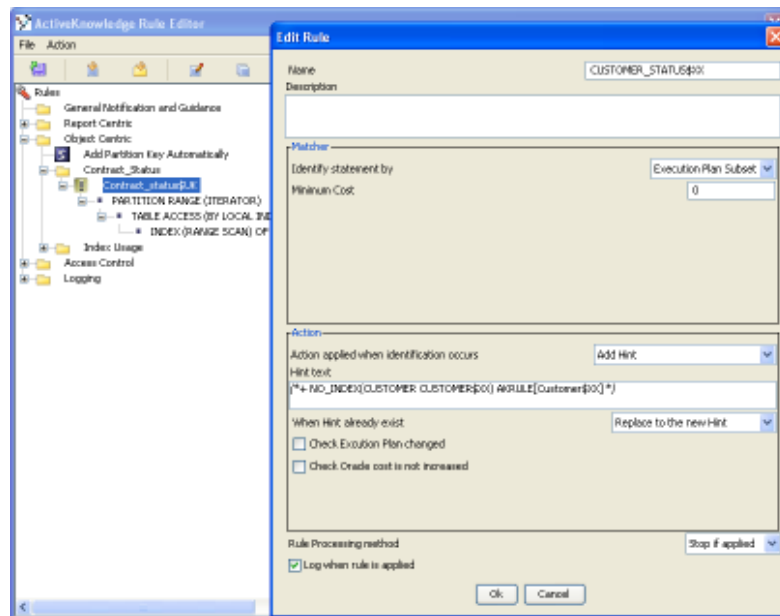
2. Performance rules saved 1,282 hours of execution time, substantially increasing database server capacity and saving resources.

3. The improvement measurements include the effect of only 12 performance acceleration rules defined in ActiveBase Performance server.

Examples of ActiveBase Performance™ rules

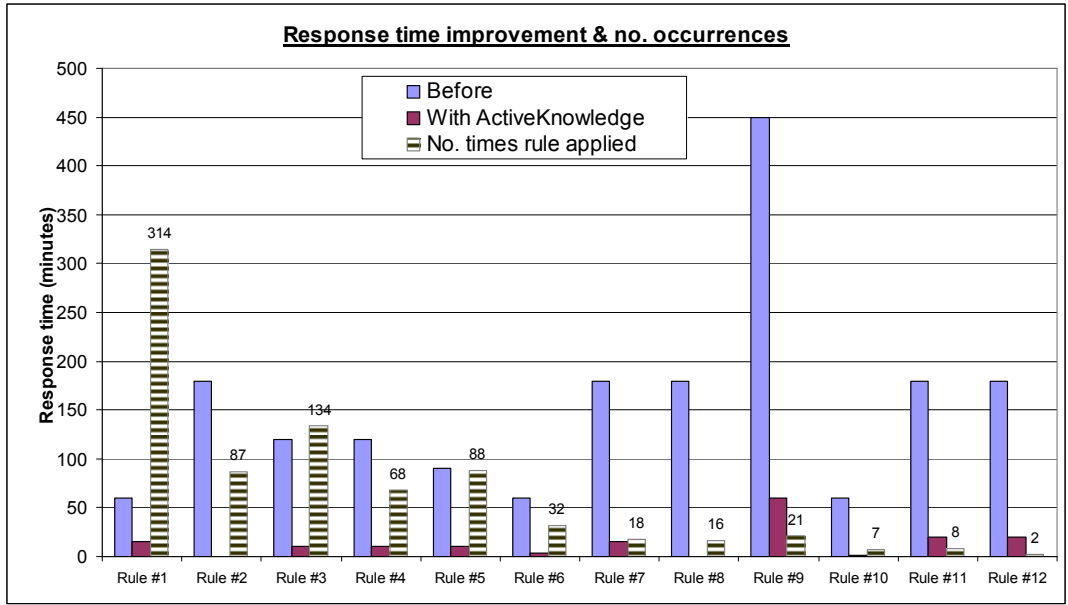
1. Reports were running on many partitions where using large index-scans instead of full-scans causing time and resource waste.

A rule was created in ActiveBase Performance with an 'Execution plan subset' matcher that identifies all reports and ad-hoc queries that execute an 'index scans' explain-plan on a partitioned table. The rule automatically adds the hint `*+ NO_INDEX*` to these SQL statements, changing the optimizer to execute using faster full scans on the partitions instead of slow massive index scans.



2. Reports and ad-hoc queries did not use an index-scan when customer_number condition appeared in the 'Where' clause. The rule used a SQL Text matcher to identify reports and ad-hoc queries when `%customer_number IN%` condition is found. Using an 'Add Hint' action, the `/*+ index (calls$customerkey) */` hint is automatically added to the reports and ad-hoc queries, enforcing the correct index usage and speeding report execution times.

3. Reports and ad-hoc queries were running on too many partitions because partition key restrictions were not properly entered (due to data format conversions) or simply missing from the 'Where' clause. The rule changed predicate time/date format to use partition key correctly using SQL 'where' clause rewrite action.



www.active-base.com



400-00101-033 | 03/08 | © 2009 ActiveBase, Ltd. All rights reserved. All other third-party trademarks are the property of their respective owners.