

# Improving Performance of a Packaged Billing Application at Leading Carrier

## Overview

A leading carrier had experienced application performance degradation due to an excessive batch processing that has exceeded 15 hours. As the batch has been running from one of the system's packaged modules, the customer was not able to touch the source-code.

**The customer has installed ActiveBase Performance™ software on the database server and within a couple of days was successful in reducing the batch process elapse time from more than 15 hours to 20 minutes.**

The improvement was made possible by using ActiveBase real-time SQL optimization rewrite rules that replace the 'from' clause object of incoming batch requests, by a much smaller and partitioned object, thus reducing execution time substantially and saving server resources.

With ActiveBase, the customer was able to complete his billing cycle in substantially less time, thus enabling the carrier to run billing batches more frequently, improving business visibility and increasing revenues.

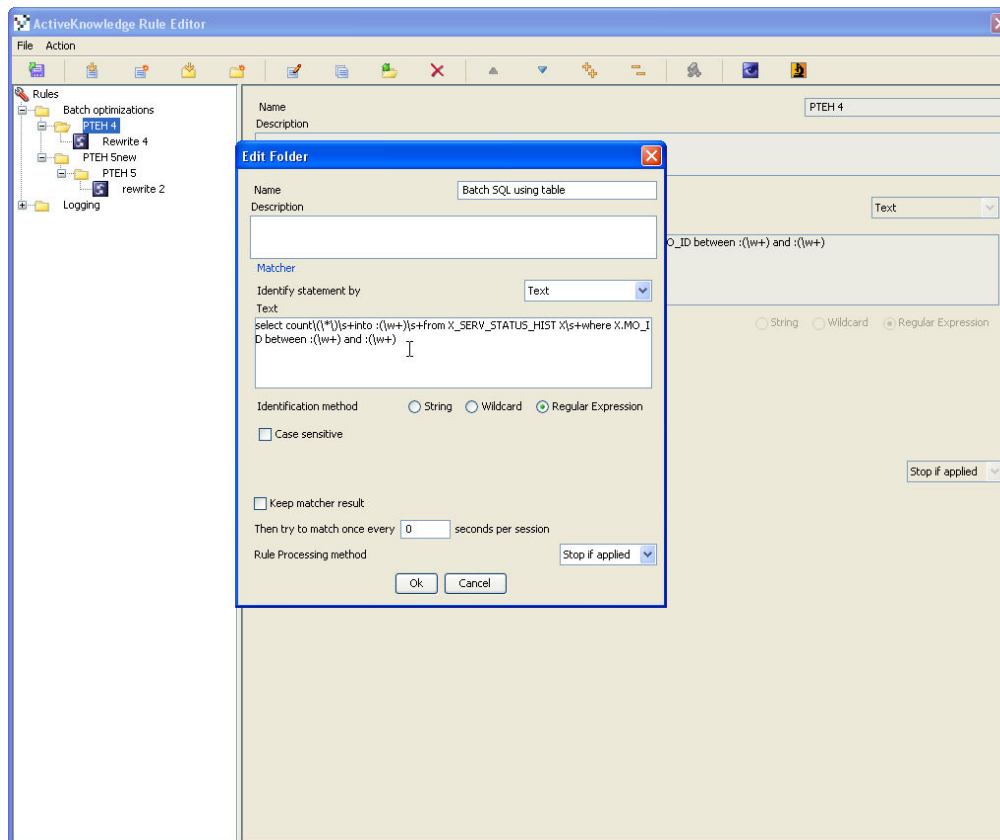


Image: ActiveBase Performance™ flexible rule tree enables to define multiple performance enhancement rules with the most complex rule-flow, to fix the most demanding performance problems

## Implementation steps

Following a short on-site training, the customer has defined several rules to fix the incoming SQL request by changing the object to much smaller and partitioned objects. These rewrites have made the performance gains reported by the customer.

**Step 1: Install ActiveBase.** ActiveBase was installed within hours on the billing database Sun enterprise Unix server. The ActiveBase listener port was configured to listen to incoming application requests through port 1526 (the database listener port was listening to port 1521).

**Step 2: Define performance optimization rules.** The customers' DBA defined several rules that identify usage of large tables in the 'select into from X\_SERV\_STATUS\_HIST...' clause (such as table X\_SERV\_STATUS\_HIST), replacing them automatically with a smaller partitioned object (X\_SERV\_STATUS\_HIST\_TEH).

Rule details:

In order to identify the batch requests with 'select ... into...', referencing the table, a regular expression rule was created with the following SQL identifying expression:

```
'select count\(\{*\}\s+into \:(\w+)\s+from X_SERV_STATUS_HIST X\s+where X.MO_ID between \:(\w+) and \:(\w+)'
```

To be replaced with the following optimized SQL request:

```
'select /*+ FULL(PST) */ count(*) into \:(1) from X_SERV_STATUS_HIST_TEH PST where X.MO_ID between \:(2) and \:(3)'
```

(for more information about regular expression and tag placeholders refer the product user guide).

**Step 3: Routing batch modules through ActiveBase Performance™ Listener port** Rules were quickly applied by simply changing the batch-module connection configuration file to connect to the database using port 1526 (instead the default port 1521). No other changes were required to other application modules or to the database itself.

**Edit Rule**

Name: Rewrite

Matcher: Text

Identify statement by: Text

Text: select count\(\{\*\}\s+into \:(\w+)\s+from X\_SERV\_STATUS\_HIST X\s+where X.MO\_ID between \:(\w+) and \:(\w+)

Identification method:  String  Wildcard  Regular Expression

Case sensitive

Keep matcher result

Then try to match once every 0 seconds per session

Action:

Action applied when identification occurs: Rewrite

Alternate statement: select /\*+ FULL(PST) \*/ count(\*) into \:(1) from X\_SERV\_STATUS\_HIST\_TEH PST where X.MO\_ID between \:(2) and \:(3)

Rule Processing method: Stop if applied

Log when rule is applied

Ok Cancel