

Installs in
minutes

GL Wand Performance Optimisation

Version 4.00

www.excel4apps.com

Excel4Apps (Pty) Ltd

support@excel4apps.com

All rights reserved.

with internet architecture



PERFORMANCE OPTIMISATION 3

- Recommended configuration..... 3
- Creating the GL_CODE_COMBINATIONS_CAT index..... 5
- Run "Gather Statistics" program 7
- Timeout Profile Option 7
- HTTP Packet Size 7
- Confirm existence or create required indexes..... 7
- Using an existing custom index 8
- Run the GL Optimiser Program..... 8
- Summary Accounts 8
- Get Balance Formulas 9
- Remote system health check 9



Performance optimisation

Most GL Wand customers experience average cell refresh rates of between 10 and 50 cells per second, though some experience 100+ cells per second refreshes. For those with refresh rates of less than 1 cell per second or with reports that timeout, this section may provide solutions for improvement.

Many factors will affect the cell refresh performance including for example the PC specification, the server age and specification, as well as other processes running on the server. The customer remains ultimately responsible for the performance of the SQL queries running on your Oracle system. However, based on our experience at many of our customers over a number of years, we can provide the following tips and suggestions which might assist to improve performance.

Firstly we will discuss the recommended configuration and then we will include all the other possible techniques that you can try.

Recommended configuration

We have found that the following configuration provides, on average, the best performance:

- ★ Running that latest GL Wand version
- ★ Parent Cache option enabled
- ★ Get Balance hint overridden to the "No Hint" option
- ★ GL_CODE_COMBINATIONS_CAT index created

GL Wand Version

At the time of publishing this document that latest version is GL Wand client 4.10 and server 4.00. We recommend that you always upgrade to the newest version as soon as you can as we are continually making changes in the new releases to improve performance.

Parent Cache Option

To enable the parent cache option navigate to GL Wand Toolbar > Tools > Options > Advanced. Tick the "Enable the parent cache option on logon form" item.



The screenshot shows the 'Options' dialog box with the following sections:

- Connections:** A table with columns 'Connection Name' and 'Connection URL'. The first row is 'Excel4apps Demo' with URL 'http://visionjhb1.excel4apps.com/pls/VIS'. There are three empty rows below it. Buttons 'Mail', 'Test', and 'Delete' are at the bottom.
- Upgrade:** A section with 'Internet' and an 'Upgrade' button.
- Navigation:** 'Basic <<', 'OK', 'Cancel', and a help icon.
- Preferences:** A list of checkboxes and dropdowns:
 - Disable Summary Account Checking
 - Enable trace mode
 - Show secured segment values
 - Budget upload - delete duplicates in the interface table
 - Enable the parent cache option on logon form
 - Enable batching on drill down requests
 - HTTP Packet Size: Dynamic
 - Logon Method: Orade
 - Batch size: 1000
 - Show disabled segment values
- Report Manager:** Process Disabled Segment Values and eMail Client: Outlook - Draft

Now log on again with GL Wand. A new option will appear on the logon form called "Cache parent values". Tick this option before you complete the logon.



Installs in
minutes



Get Balance Hint

By default GL Wand will include a hint in the SQL statement to use the `GL_CODE_COMBINATIONS_CAT` index if it exists. In older releases of Oracle this consistently provided the best performance and was therefore made the default behaviour. However in the newer Oracle releases it is not always the best option and therefore we recommend that you disable this option as follows:

- ★ Log on to Oracle with the System Administration responsibility
- ★ Navigate to Profile > System
- ★ Find all profile options that start with "GL Wand%"
- ★ In the option called GL Wand: Get Balance Hint enter the following:

```
/* */
```

- ★ Save the option.
- ★ Log on again with GL Wand for this change to take effect.

GL_CODE_COMBINATIONS_CAT index

The creation of this index can significantly improve performance. The detail of exactly how to create this index is set out in the next section.

Creating the `GL_CODE_COMBINATIONS_CAT` index

The process to create this index is as follows:

1. Determine the number of distinct values per segment in your `GL_CODE_COMBINATIONS` table

with internet architecture



2. Order your segments from largest (most distinct values) to smallest
3. Determine the most frequently used segments for General Ledger Reporting.
4. Create the GL_CODE_COMBINATIONS_CAT index specifying the order of the segments as follows:
5. As a general rule, order your segments from largest (most distinct values) to smallest
6. If however your chart of accounts contains some segments which contain many distinct values but are used infrequently for reporting, then you should rather include the frequently used segments first in the index.

To determine the order of an index execute the following select statement:

```
SELECT COUNT(DISTINCT &segment_name) FROM gl_code_combinations;
```

This select statement counts the number of distinct values for a column name that you enter for the GL_CODE_COMBINATIONS table. Enter SEGMENT1, rerun with SEGMENT2, and for all segments until the last segment used in your account flexfield (SEGMENT6 in our example).

In our **example** the select statement could return the values:

- ★ COUNT(DISTINCT SEGMENT1) : 214
- ★ COUNT(DISTINCT SEGMENT2) : 355
- ★ COUNT(DISTINCT SEGMENT3) : 1134
- ★ COUNT(DISTINCT SEGMENT4) : 1255
- ★ COUNT(DISTINCT SEGMENT5) : 21
- ★ COUNT(DISTINCT SEGMENT6) : 5423

From the data above, the best order to make the most selective index is:

- SEGMENT6, SEGMENT4, SEGMENT3, SEGMENT2, SEGMENT1, SEGMENT5

However, in our example segment 4 is the account segment and is used most in reporting. Segment 6 is a project segment and has many values but is not used much in reporting. The revised index order would then be:

- SEGMENT4, SEGMENT6, SEGMENT3, SEGMENT2, SEGMENT1, SEGMENT5

It worthwhile to compare performance of the above mentioned order to an index that excludes segment 6 completely. Especially if segment 6 is never used for reporting.

Create GL_CODE_COMBINATIONS_CAT with the concatenated index as below:

In our **example** you should sign on to SQL*Plus as APPS and execute the following:

(Replace this with your own SQL statement)

```
CREATE INDEX gl_code_combinations_cat ON gl_code_combinations (segment4,  
segment6, segment3, segment2, segment1, segment5);
```

This can be a process of trial and error. You might find that your system provides better performance with an index order slightly different from the recommendation. It is therefore



worthwhile to try a few variations in order to identify the order that works best on your system.

The creation of this index is recommended by Oracle in certain circumstances. Refer to metalink note ([198437.1](#)).

Run "Gather Statistics" program

The cost based optimiser will attempt to process SQL statements as efficiently as possible. In order for the Cost Based Optimiser to work effectively the "Gather Statistics" program needs to be executed on a regular schedule. Refer to the applicable Oracle documentation for more information on the "Gather Statistics" program.

Based on experience with our customers we recommend the following gather statistics percentages:

Table	Percentage	Comment
GL_BALANCES	30%	Recommended minimum
GL_CODE_COMBINATIONS	100%	Depending on the size of this table, but seeing as this table is used extensively by all GL Wand SQL statements 100% is recommended even if the table is fairly large.
FND_FLEX_VALUES	100%	
FND_FLEX_VALUE_HIERARCHIES	100%	

Timeout Profile Option

If the users are experiencing timeouts during GL Wand report execution ensure that the GL Wand profile options are set appropriately. Please see the user guide for details on how to do this.

HTTP Packet Size

Try different settings for the HTTP packet size (GL Wand Options). This can also assist where users are experiencing timeouts during GL Wand report execution. Please see the user guide for details on how to do this.

Confirm existence or create required indexes

GL Wand is designed to utilise the following indexes to improve performance of the queries. These indexes are either standard or have been recommended by Oracle to improve reporting performance.

We recommend that you ensure that these are created.

Table	Index	Comment
GL_BALANCES	GL_BALANCES_N1	Standard index
GL_CODE_COMBINATIONS	GL_CODE_COMBINATIONS_N1 TO _NX	Created by the GL Optimiser program.



FND_FLEX_VALUES	FND_FLEX_VALUES_N1	Standard index
FND_FLEX_VALUE_HIERARCHIES	FND_FLEX_VALUE_HIERARCHIES_N1	Standard index
GL_CODE_COMBINATIONS	GL_CODE_COMBINATIONS_CAT	This custom index can be created manually to improve performance in certain circumstances.

Using an existing custom index

If your Oracle system already has a custom index with concatenated segment columns then GL Wand can take advantage of this index. You can use the Get Balance Hint profile option to create a custom SQL hint in the Get Balance SQL that will use your custom index.

We recommend that you test a GL Wand Get Balance SQL statement and analyse the explain plan before and after the introduction of this custom hint to assess what impact the hint has on the execution path. Please contact support@excel4apps.com for assistance with the process.

Run the GL Optimiser Program

GL Optimiser is a standard GL concurrent request. It runs the gather statistics program in the GL schema and creates indexes for your accounting flexfield segment columns. Refer to your Oracle documentation for additional information.

Summary Accounts

For a small percentage of our customers and only after the above steps have still not had the desired effect, the creation of summary accounts may provide the desired performance gains.

The Oracle General Ledger provides the ability to create summary accounts which are special accounts that hold a balance at a summary level. GL Wand can use these summary accounts to facilitate faster reporting. Typically, summary accounts will help with reporting performance where the reports run at higher levels in the reporting hierarchy take very long to complete. For example, if you have a cost centre hierarchy with many levels and running a GL Wand report at the top level cost centre takes very long, you could create summary accounts at this top level cost centre and this could dramatically improve reporting time. The process is as follows:

- ★ Decide at what level in the hierarchy the GL Wand reports will be executed.
- ★ Follow the instructions in the Oracle General Ledger user guide to create summary accounts at this level.

Performance gains can be quite substantial but it will depend on your specific configuration and reporting requirements. Below is a case study of the performance gains that have been achieved:

Customer A had the following

with internet architecture



- ★ 15 million GL Balance records
- ★ 8 segments in the accounting flexfield
- ★ Complex cost centre hierarchy with many levels
- ★ GL Wand report only reported on values in the first 3 segments
- ★ GL Wand report had 1900 Get Balance calculations
- ★ Report was running for more than 20 minutes

After creating summary accounts the report running at the top level cost centre ran in 55 seconds.

To configure GL Wand to use summary accounts go to the GL Wand options form and uncheck the "Disable Summary Account Checking" option, which is set by default.

Get Balance Formulas

When creating Get Balance formulas always use "%" to denote all values as this allows more efficient SQL to be generated. Do not use a range like "000-ZZZ" or a parent like "T" as they will require unnecessary processing.

Remote system health check

If you are still having performance issues then please log a performance related case by clicking the support button or emailing support@excel4apps.com. We offer a complimentary remote tuning service to assist with this process.

