

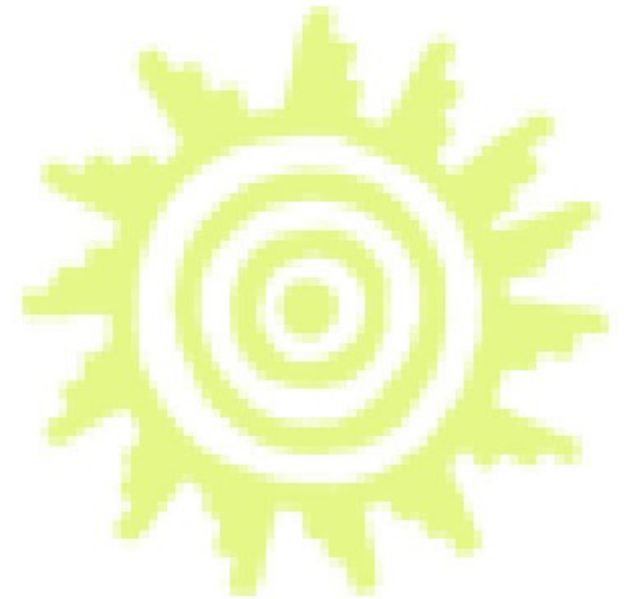
Using Oracle Application Express to analyze your PL/SQL source code

Oracle Open World 2008

September 22, 4pm

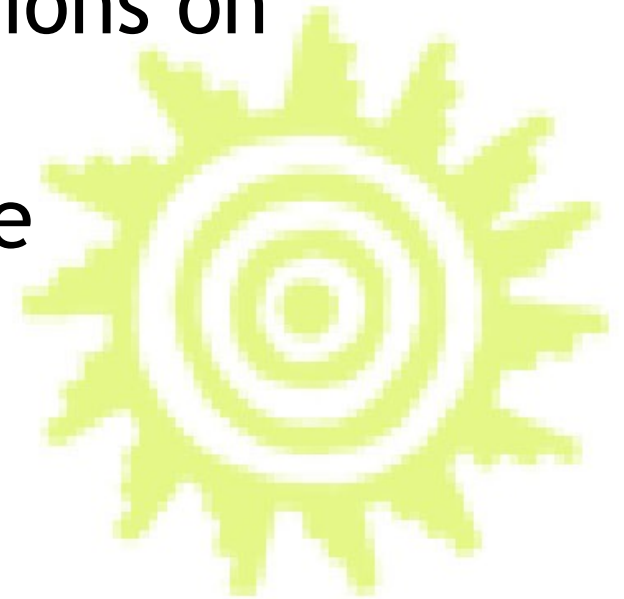
Room 236

Moscone South



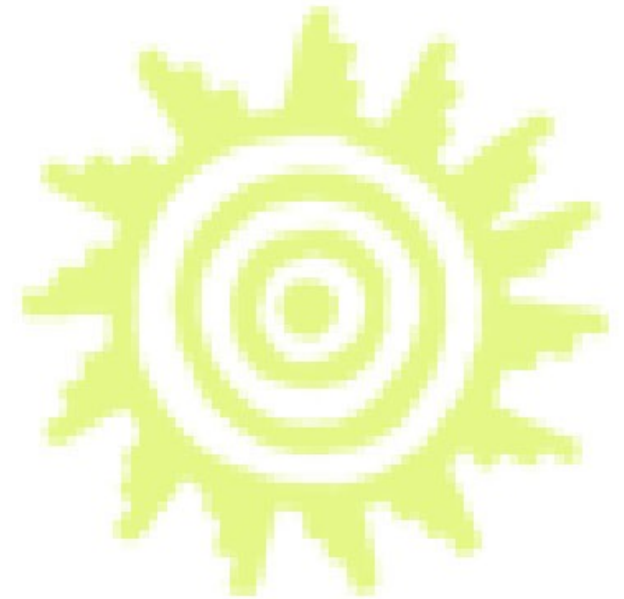
Some facts about me

- Flavio Casetta
- Founder of Yocoya.com
- Editor of blog OracleQuirks.blogspot.com
- 25+ years in the IT
- 10+ years developing applications on Oracle
- 15+ years developing database applications
- 4+ years on Apex



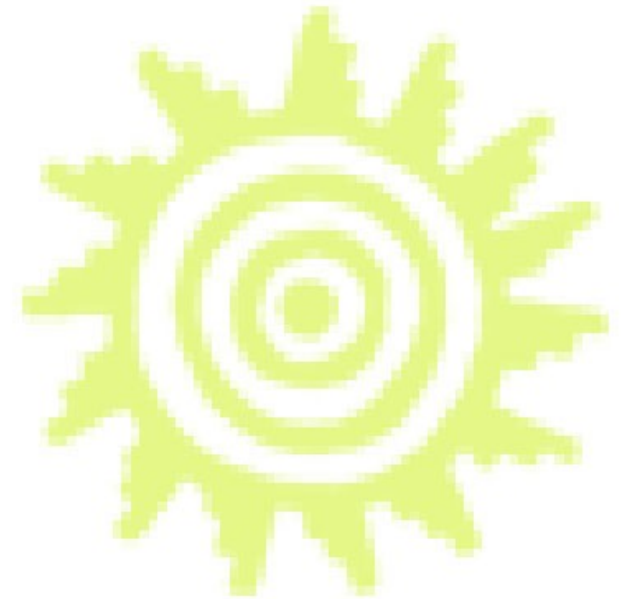
Agenda

- Oracle Application Express does it better
- The power of DBMS_PROFILER
- The happy marriage of Apex and Profiler
- A real life scenario
- Conclusions
- Q&A



Why Oracle Application Express ?

- It comes with the RDBMS
- Supports Oracle 9R2 +
- Quick development cycle
- Multi-user
- It's free
- Apex it's cool!



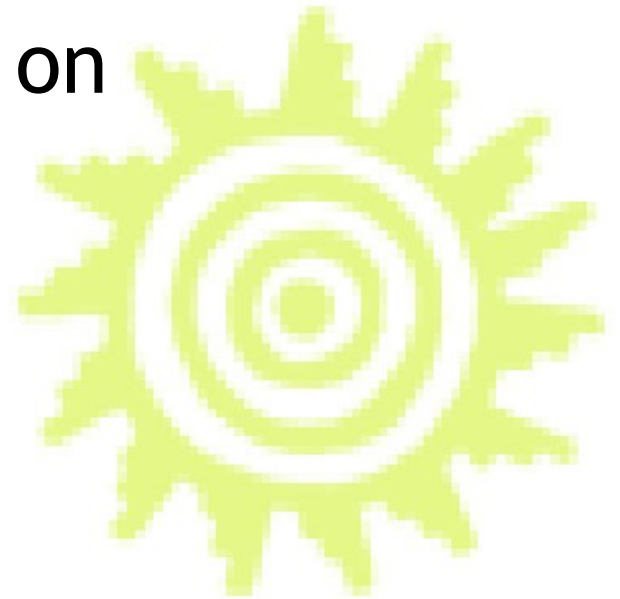
Built-in PL/SQL Profiler

- Built-in package (but requires tables set up) before use.
- Very easy to install and set up, slightly less straightforward to interpret the results
- Tracks the performance of each PL/SQL statement
- Supports the concept of “run”
- Supports some degree of customization
- Small overhead
- Can serve as a basis for further analysis



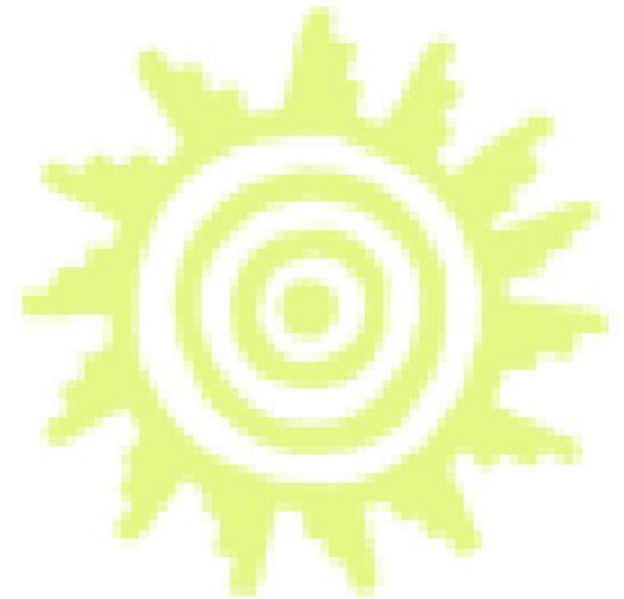
Celebrating the happy marriage between Apex and the profiler

- (almost) Point and click installation
- No need to load the script from the command line
- Does some manual operations on your behalf
- Gets the job done quickly and easily!
- It's free!



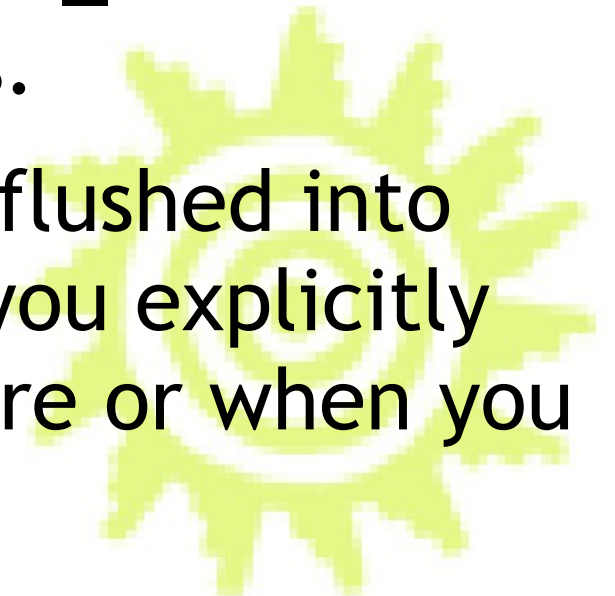
Extensions to the base profiler

- Dedicated repository
- Concept of PL/SQL source code snapshots
- Off-line analysis



How does the profiler work?

- The PL/SQL profiler supports this process using the concept of a "run".
- The beginning and the ending of a run is controlled by calling the `START_PROFILER` and `STOP_PROFILER` functions.
- Collected data for the run are flushed into database tables either when you explicitly call the `FLUSH_DATA` procedure or when you stop the profiler.



How do we start collecting data?

```
begin
```

```
  DBMS_PROFILER.START_PROFILER(
```

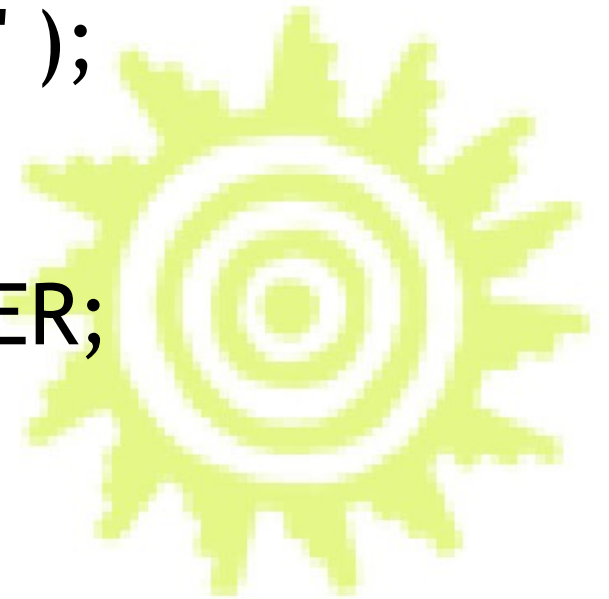
```
    run_comment => 'test procedure',
```

```
    run_comment1 => 'test set #1' );
```

```
  /* call your code here */
```

```
  DBMS_PROFILER.STOP_PROFILER;
```

```
end;
```



Schema Contents

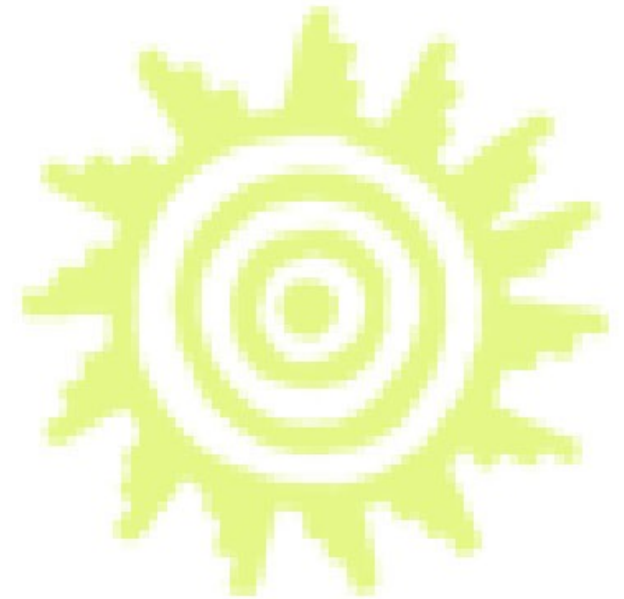
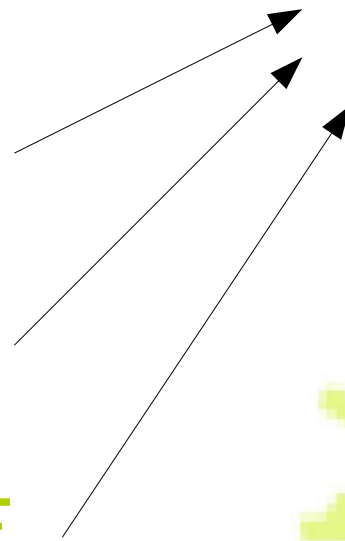
owner packages

- PROFI YY_PKG_PROFILER

tables

- PLSQL_PROFILER_RUNS
- PLSQL_PROFILER_UNITS
- PLSQL_PROFILER_DATA
- PLSQL_PROFILER_SOURCE
- PLSQL_PROFILER_SOURCE_LINES

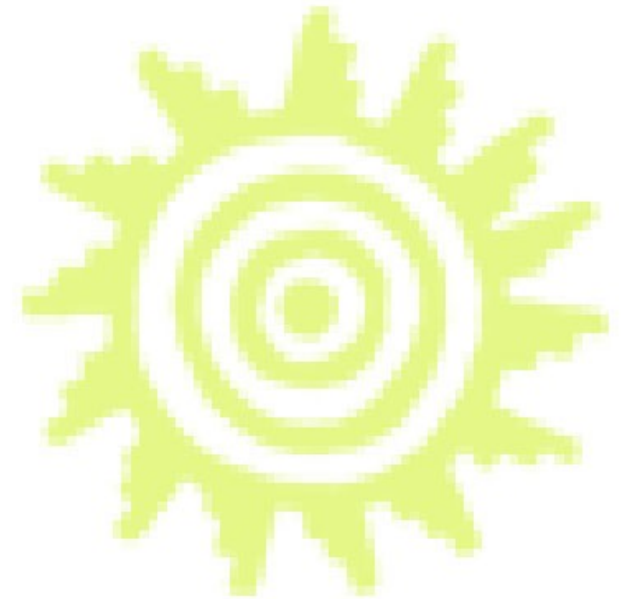
synonyms



PLSQL PROFILER RUNS

Master table containing:

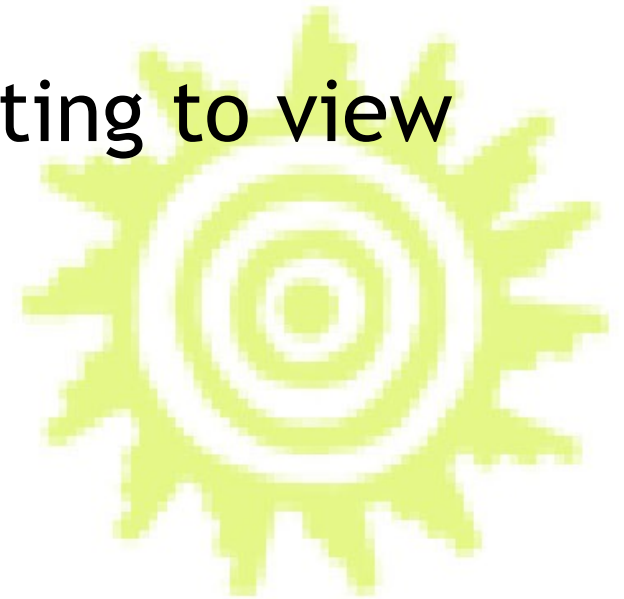
- overall execution time
- run identifier
- user comments
- timestamp



PLSQL PROFILER UNITS

Detail table containing summary data about each source program found by the profiler

- Unique Identifier for each source object
- Total execution time for each unit (to be calculated off-line)
- References to the source pointing to view `ALL_SOURCE`



PLSQL PROFILER DATA

Detail table containing the execution time of each and every line of code executed by the profiler:

- Unique Identifier
- Number of times the line was executed
- Total execution time for each source line
- Maximum and minimum duration
- References to the other tables



PLSQL PROFILER SOURCE

Master table containing relevant information about the source code:

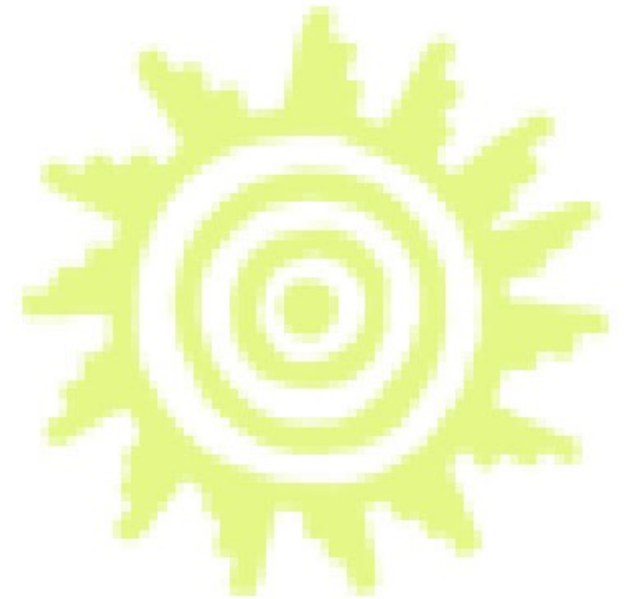
- Key to uniquely identify the source code
- Eliminate redundant copies of the source
- The timestamp allows to compare the same unit over the time



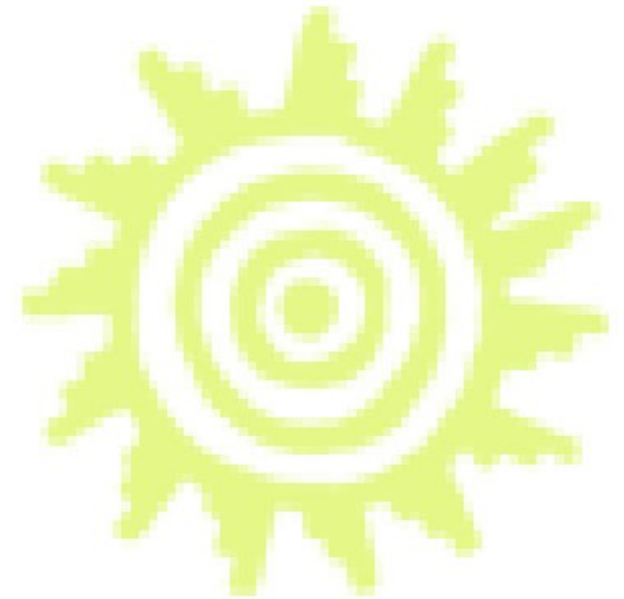
PLSQL PROFILER SOURCE LINES

Detail table containing a snapshot of the source code lines:

- columns are taken from ALL_SOURCE upon completion of the execution.



How do you know when it's the time to run the profiler?



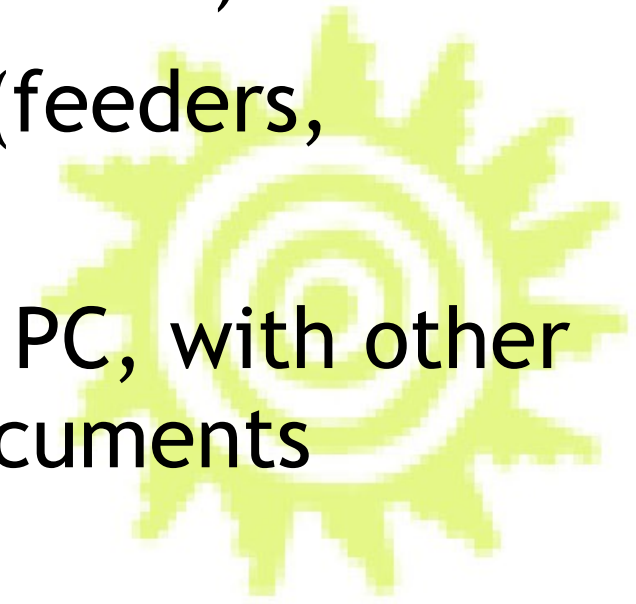
My customer's case

My customer is the largest distributor of newspapers in the Netherlands

Distribution plant is made up of 2 distinct lines (run by the same db instance)

Each line features 27 stations (feeders, printers, palletizers)

Each line is supervised by a PC, with other PCs as clients for printing documents



Apex profiler screenshot #1

Summary Runs Units

Summary

summary of records by owner and time scale

Schema	Time Scale ▲	Runs	Earliest Date	Latest Date
INFOVISION	milliseconds (ms)	185	28/05/08 10:22:51	28/05/08 19:28:12
INFOVISION	tenths of ms	12467	28/05/08 10:22:47	28/05/08 19:38:38
INFOVISION	hundreds of ms	30	28/05/08 12:36:12	28/05/08 19:23:06
INFOVISION	seconds	4	28/05/08 14:17:42	28/05/08 18:41:58
YOCOYA	seconds	6	17/08/08 00:11:35	20/08/08 12:29:42

1 - 5

runs in chronological order

Interval Start Interval End

Schema	Comment	Additional Comment	Duration	Run Date ▼	Units
YOCOYA	test 6	-	4.915s	20/08/08 12:29:42	6
YOCOYA	test 5	-	5.049s	20/08/08 12:06:42	6
YOCOYA	test 4	-	1.094s	17/08/08 00:13:07	6
YOCOYA	test 3	-	1.054s	17/08/08 00:12:44	6
YOCOYA	test 2	-	1.082s	17/08/08 00:12:20	6
YOCOYA	test 1	-	1.206s	17/08/08 00:11:35	6
INFOVISION	22	-	0.009s	28/05/08 19:38:38	0
INFOVISION	22	-	0.009s	28/05/08 19:37:52	0
INFOVISION	22	-	0.013s	28/05/08 19:37:48	0
INFOVISION	22	-	0.010s	28/05/08 19:37:46	0
INFOVISION	22	-	0.008s	28/05/08 19:37:44	0
INFOVISION	22	-	0.009s	28/05/08 19:37:41	0
INFOVISION	22	-	0.009s	28/05/08 19:37:39	0
INFOVISION	22	-	0.011s	28/05/08 19:37:36	0
INFOVISION	22	-	0.010s	28/05/08 19:37:34	0

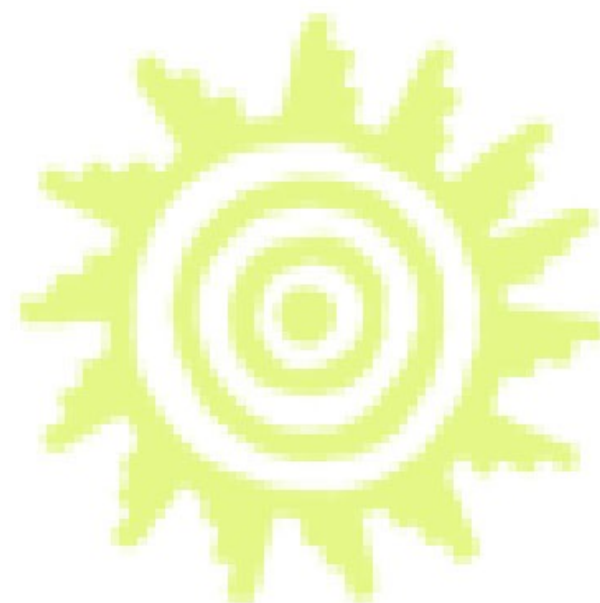
1 - 15 Next >

settings

Date Format

Rows

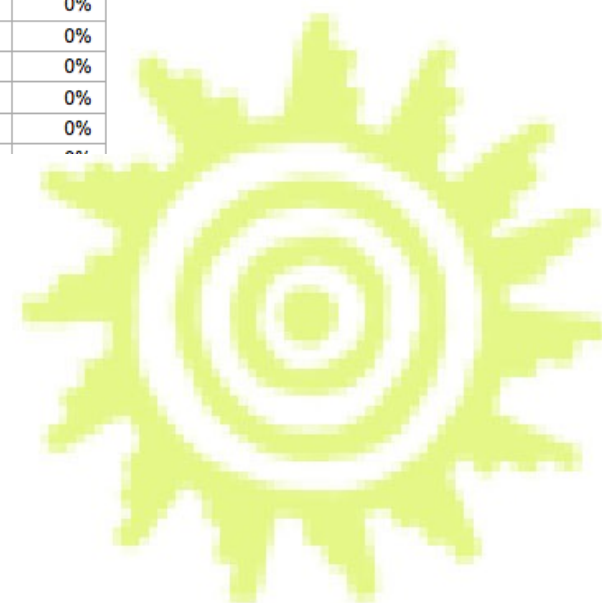
press [ENTER] to submit changes



YOCOYA

Apex profiler screenshot #2

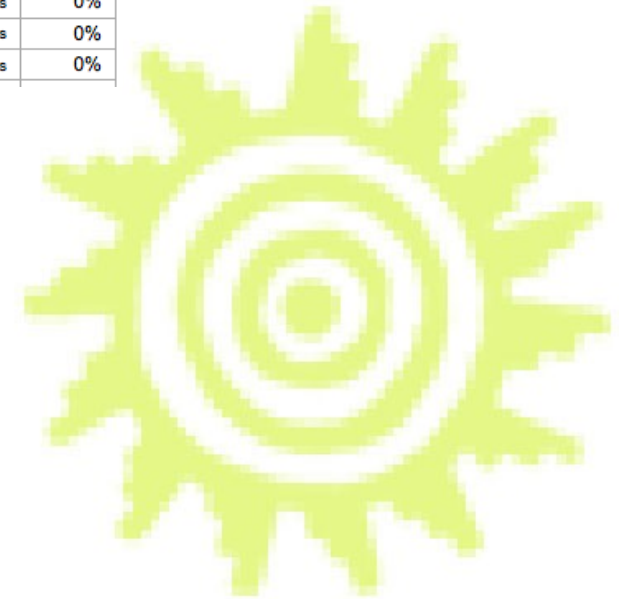
If num_stn = info_const.conReordering_Station then	213	1	0.000s	0.000s	0.000s	0%
	214	-	0.000s	0.000s	0.000s	0%
tote_cnt := content_varry(position(null, null, null),	215	1	0.000s	0.000s	0.000s	0%
position(null, null, null),	216	-	0.000s	0.000s	0.000s	0%
position(null, null, null),	217	-	0.000s	0.000s	0.000s	0%
position(null, null, null),	218	-	0.000s	0.000s	0.000s	0%
position(null, null, null),	219	-	0.000s	0.000s	0.000s	0%
position(null, null, null),	220	-	0.000s	0.000s	0.000s	0%
position(null, null, null),	221	-	0.000s	0.000s	0.000s	0%
position(null, null, null),	222	-	0.000s	0.000s	0.000s	0%
position(null, null, null),	223	-	0.000s	0.000s	0.000s	0%
position(null, null, null));	224	-	0.000s	0.000s	0.000s	0%
	225	-	0.000s	0.000s	0.000s	0%
For each_OOF_position in OOF_station(expected_tote_id)	226	3	0.997s	0.000s	0.997s	99%
Loop	227	-	0.000s	0.000s	0.000s	0%
tote_cnt(each_OOF_position.pos).mat_id := each_OOF_position.mat_id;	228	1	0.000s	0.000s	0.000s	0%
tote_cnt(each_OOF_position.pos).lot := each_OOF_position.lot;	229	1	0.000s	0.000s	0.000s	0%
tote_cnt(each_OOF_position.pos).cps := each_OOF_position.cps; -- assumi...	230	1	0.000s	0.000s	0.000s	0%
End loop;	231	-	0.000s	0.000s	0.000s	0%



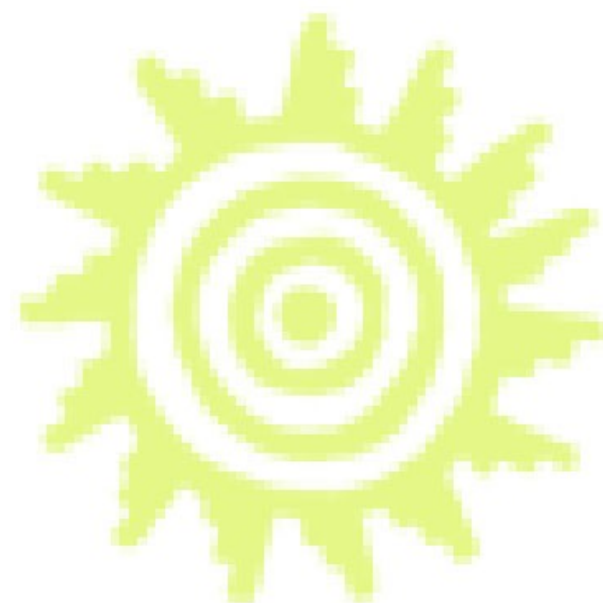
Apex profiler screenshot #3

- Profiler having problems with triggers!

	29	-	0.000s	0.000s	0.000s	0%
If :new.lst_of_cst = info_const.conYes then	30	0	0.000s	0.000s	0.000s	0%
-- case of a box...	31	-	0.000s	0.000s	0.000s	0%
Kill_Print(err_code, err_msg, :new.pkg_sht_id, prt_stn);	32	1	0.000s	0.000s	0.000s	0%
	33	-	0.000s	0.000s	0.000s	0%
If err_code <> 0 then	34	-	0.000s	0.000s	0.000s	0%
:new.sts := prt_stn;	35	1	0.000s	0.000s	0.000s	48%
:new.inv_wgt := err_code; -- logging the p...	36	-	0.000s	0.000s	0.000s	0%
End if;	37	-	0.000s	0.000s	0.000s	0%
	38	-	0.000s	0.000s	0.000s	0%
End if;	39	-	0.000s	0.000s	0.000s	0%



Q&A



September 22, 2008

Oracle Open World 2008, Session 301401