BETTER DATE AND ELAPSED TIME REPORTING FORMATS FOR BUSINESS USERS

I am a business user, so why say to me "it ran for 0.049 days"?

All too often I see reports created by technical staff and DBAs being presented to management and business users with elapsed times (such as concurrent request run times) calculated in days or seconds and this often causes frustration amongst report recipients. Here are some typical examples:-

- A concurrent request run time of 1 hour and 10 minutes is commonly reported as:

❖ 0.0486 days or
❖ 4,200 seconds

- An incident report that states the issue occurred 20-Sep-10 12:24:23. What is often more helpful to the user is to understand the day of the week of the incident; was that a Monday (our high processing day) or a Tuesday? And seeing the seconds reported usually adds no value at all…

The bottom line is good reporting should mean a report user does not have to perform mental gymnastics around dates and elapsed time calculations.

Preferred Date Format

Let's start with something simple, the standard date format - usually in the format:

```
SELECT sysdate FROM dual;
```

Depending on your default date format you will get something like the following:

```
29-Sep-2010 1:01:57 PM
```

Whilst you may think this is OK, with this date format the business user has to consult the calendar to work out what day of the week the report refers to; "was the 29th a Wednesday or Thursday"? Remember in the business world different business processes occur on different days. When presenting dates in management or business user reports you should add the day value to the date format:

```
SELECT to_char( sysdate, 'DD-Mon-YY (Dy) HH24:MI' ) FROM dual;
```

This provides the more useful date format:

```
29-Sep-10 (Wed) 13:04
```

In general, when reporting on business activities management and business users don't care about seconds, minutes are close enough.

Preferred elapsed time format

When working with dates it is very easy to calculate the elapsed time between two dates (such as a concurrent request run time) by subtracting two dates (*actual_completion_date* - *actual_start_date*) and presenting the elapsed time in days or in seconds by multiplying the two subtracted dates by 24 * 60 * 60…

Example:

A concurrent request run time of 1 hour and 10 minutes is often reported as:

- 0.0486 days or
- 4,200 seconds

A practical example of generating elapsed time in this format is:

```
SELECT fcr.request_id,
       fu.user_name,
       fcpt.user_concurrent_program_name,
       round((fcr.actual_completion_date-fcr.actual_start_date),2) "Run Time Days",
       round(((fcr.actual_completion_date-fcr.actual_start_date)
              * 24*60*60),2)"Run Time Seconds"
  FROM applsys.fnd_concurrent_requests fcr,
       applsys.fnd_user fu,
       applsys.fnd_concurrent_programs_tl fcpt
 WHERE fcr.requested_by = fu.user_id
   and fcr.program_application_id = fcpt.application_id
   and fcr.concurrent_program_id = fcpt.concurrent_program_id
   and fcpt.language = userenv('LANG')
   and fcr.actual_start_date is not null
   and fcr.actual_completion_date is not null;
```

What you should remember when dealing with management or users is they usually hate having to perform mental gymnastics to calculate a time, what is 0.02 days or 1,454 seconds? Would it not be better if the time was quoted as 28 minutes? I believe it's always preferable when presenting information to management and business users to convert elapsed time to Days, Hours and Minutes.

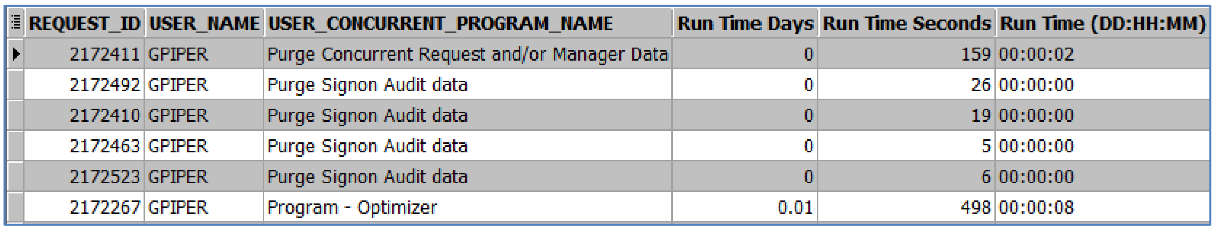
There are a number of ways to generate the elapsed time in Days:Hours:Minutes: here is just one:

```
nvl(ltrim(to_char(trunc(sysdate  - working_date_value),            '99900')),0) ||':'|| -- Days
nvl(ltrim(to_char(mod(trunc((sysdate - working_date_value)*24),24),    '00')),0) ||':'|| -- Hours
nvl(ltrim(to_char( mod(trunc((sysdate - working_date_value)*1440),60), '00')),0) Minutes,
```

Adding this to the example SQL above:

```
SELECT fcr.request_id,
       fu.user_name,
       fcpt.user_concurrent_program_name,
       round((fcr.actual_completion_date-fcr.actual_start_date),2) "Run Time Days",
       round(((fcr.actual_completion_date-fcr.actual_start_date)
              * 24*60*60),2) "Run Time Seconds",
       nvl(ltrim(to_char(trunc(fcr.actual_completion_date - fcr.actual_start_date),
             '99900')),0) ||':'|| -- Days
       nvl(ltrim(to_char(mod(trunc((fcr.actual_completion_date - fcr.actual_start_date)
            *24),24), '00')),0) ||':'|| -- Hours
       nvl(ltrim(to_char(mod(trunc((fcr.actual_completion_date - fcr.actual_start_date)
            *1440),60),'00')),0) "Run Time (DD:HH:MM)"
  FROM applsys.fnd_concurrent_requests fcr,
       applsys.fnd_user fu,
       applsys.fnd_concurrent_programs_tl fcpt
 WHERE fcr.requested_by = fu.user_id
   and fcr.program_application_id = fcpt.application_id
   and fcr.concurrent_program_id = fcpt.concurrent_program_id
   and fcpt.language = userenv('LANG')
   and fcr.actual_start_date is not null
   and fcr.actual_completion_date is not null;
```

Example output (right column is the preferred elapsed time reporting format)



REQUEST_ID	USER_NAME	USER_CONCURRENT_PROGRAM_NAME	Run Time Days	Run Time Seconds	Run Time (DD:HH:MM)
2172411	GPIPER	Purge Concurrent Request and/or Manager Data	0	159	00:00:02
2172492	GPIPER	Purge Signon Audit data	0	26	00:00:00
2172410	GPIPER	Purge Signon Audit data	0	19	00:00:00
2172463	GPIPER	Purge Signon Audit data	0	5	00:00:00
2172523	GPIPER	Purge Signon Audit data	0	6	00:00:00
2172267	GPIPER	Program - Optimizer	0.01	498	00:00:08

What if the report runs in less than 1 minute?

Years of experience tells me the typical answer to this from a business user is **who cares**. With anything to do with concurrent request activity, if it runs in less than 1 minute it's great ☺. Whilst DBAs operate in milliseconds and seconds, business users generally don't care about requests that run in less than a minute. Remember, for concurrent requests the sleep time for the standard manager (which runs most requests) should be 60 seconds, so on average it will take the managers 30 seconds to even start running a report.

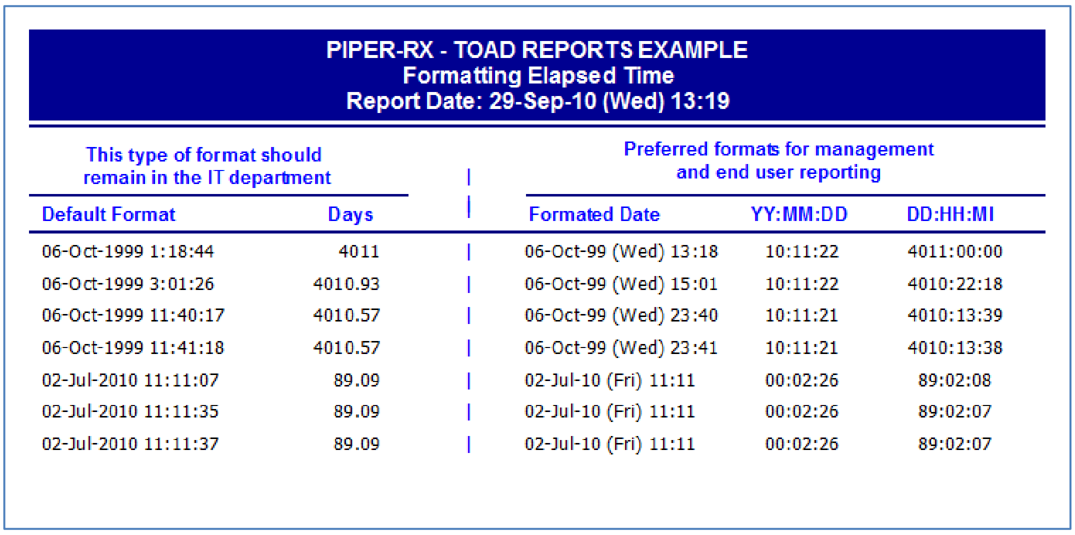
What if I have elapsed time in the years?

Where you have longer run times, usually associated with workflows that have been hanging around for years or users who have not used their accounts for years (Aged users), you should convert the elapsed time into Years : Months : Days

Again, there are any number of methods to obtain the elapsed time, here is just one:

```
lpad(trunc((months_between(sysdate, your_date)) / 12 ), 2, '0') ||':'|| -- Years
lpad(trunc((months_between(sysdate, your _date)) -
(trunc((months_between(sysdate, your _date)) / 12 ) * 12 )), 2, '0') ||':'|| -- Months
lpad(trunc(((months_between(sysdate, your _date)) -
(trunc((months_between(sysdate, your _date)) / 12 )  * 12 )
-
(trunc((months_between(sysdate, your _date)) -
(trunc((months_between(sysdate, your _date))  / 12 ) * 12 ))))
*  to_char(last_day(sysdate), 'DD')), 2, '0') "YY-MM-DD", -- Days
```

The following is an example report showing both YY:MM:DD and DD:HH:MI formats:


PIPER-RX - TOAD REPORTS EXAMPLE
Formatting Elapsed Time
Report Date: 29-Sep-10 (Wed) 13:19

This type of format should remain in the IT department			Preferred formats for management and end user reporting		
Default Format	**Days**	|	**Formated Date**	**YY:MM:DD**	**DD:HH:MI**
06-Oct-1999 1:18:44	4011	|	06-Oct-99 (Wed) 13:18	10:11:22	4011:00:00
06-Oct-1999 3:01:26	4010.93	|	06-Oct-99 (Wed) 15:01	10:11:22	4010:22:18
06-Oct-1999 11:40:17	4010.57	|	06-Oct-99 (Wed) 23:40	10:11:21	4010:13:39
06-Oct-1999 11:41:18	4010.57	|	06-Oct-99 (Wed) 23:41	10:11:21	4010:13:38
02-Jul-2010 11:11:07	89.09	|	02-Jul-10 (Fri) 11:11	00:02:26	89:02:08
02-Jul-2010 11:11:35	89.09	|	02-Jul-10 (Fri) 11:11	00:02:26	89:02:07
02-Jul-2010 11:11:37	89.09	|	02-Jul-10 (Fri) 11:11	00:02:26	89:02:07

SQL for this report is as follows:

```
SELECT to_char(sysdate, 'DD-Mon-YY (Dy) HH24:MI') report_date,
       begin_date default_date_format,
       to_char(begin_date, 'DD-Mon-YY (Dy) HH24:MI') formatted_date,
       round((sysdate-begin_date), 2) days,
       lpad(trunc((months_between(sysdate, begin_date)) / 12), 2 , '0') ||':'|| -- Years
       lpad(trunc((months_between(sysdate, begin_date)) -
           (trunc((months_between(sysdate, begin_date)) / 12) * 12)), 2, '0') ||':'|| -- Months
       lpad(trunc(((months_between(sysdate, begin_date)) -
           (trunc((months_between(sysdate, begin_date)) / 12) * 12)
            -
           (trunc((months_between(sysdate, begin_date)) -
           (trunc((months_between(sysdate, begin_date)) / 12) * 12))))
           *  to_char(last_day(sysdate), 'DD')), 2, '0') "YY-MM-DD",-- Days
       ltrim(to_char(trunc(sysdate - begin_date), '99900')) ||':'|| -- Days
       ltrim(to_char(mod(trunc((sysdate - begin_date)* 24), 24), '00')) ||':'|| -- Hours
       ltrim(to_char(mod(trunc((sysdate - begin_date) * 1440), 60) ,'00')) "DD-HH-MM"-- Minutes
  FROM applsys.wf_items;
```

Remember; always make life easier for your target user. They will appreciate it and in turn will be more likely to be supportive when you need it!

Want to know more?

There is loads more ***FREE*** information on this topic and all aspects of OEBS Application Administration at the **PIPER-Rx** website. After over 20+ years working with Oracle (the product, not the Company) and Oracle E-Business Suite (since Release 5) I have visited countless sites and pretty much seen it all when it comes to Applications Administration. Since the late 1990's I have spent more time sharing these learnings and the most popular papers and case studies I have presented are available at the **PIPER-Rx**.com website as well as a whole host of Tips and Reports I have used throughout my career.

All information at the **PIPER-Rx**.com website is ***FREE*** so why not check it out….I hope you find it useful! **– 30,000+ downloaders to date can't be wrong!**

Disclaimer