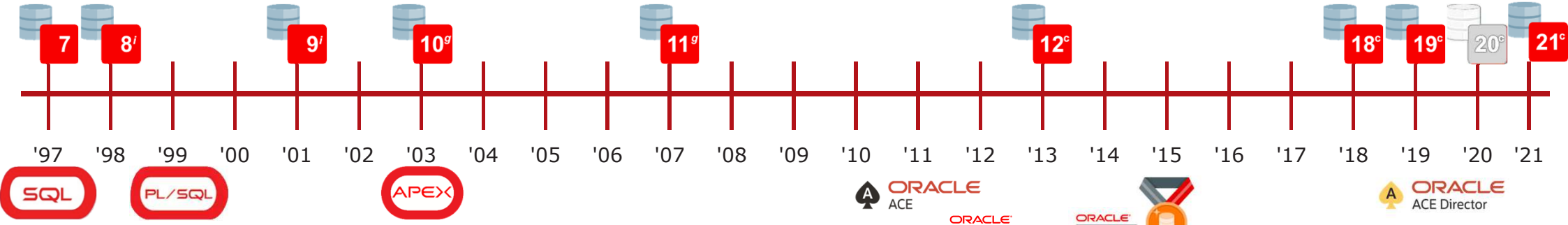
QUALOGY

PBarel@Qualogy.com
http://blog.bar-solutions.com

QUALOGY

About me…



bar-solutions.com blog http://blog.bar-solutions.com

All things ORACLE http://allthingsoracle.com

OTECH MAGAZINE http://www.otechmag.com

www.red-gate.com/simple-talk/author/patrick-barel/

bar-solutions.com/otechmagazine.php

Plugins for PL/SQL Developer
http://plugins.bar-solutions.com

Contact me…

@patch72

PBarel@Qualogy.com

Patrick.Barel@GMail.com

patrick@bar-solutions.com

Patrick.Barel@GMail.com

3029156

40338721

Patrick Barel

Oracle ACE Program

500+ technical experts helping peers globally

The **Oracle ACE Program** recognizes and rewards community members for their technical and community contributions to the Oracle community

3 membership tiers

Oracle ACE Director | Oracle ACE Pro | Oracle ACE Associate

Oracle ACE

Nominate

~~yourself~~ or someone you know:

ace.oracle.com/nominate

For more details on Oracle ACE Program:
ace.oracle.com

Connect: **aceprogram_ww@oracle.com** Facebook.com/OracleACEs @oracleace

Oracle Cloud Infrastructure

New Free Tier

oracle.com/cloud/free

Always Free

Services you can use for unlimited time

+

30-Day Free Trial

Free credits you can use for more services

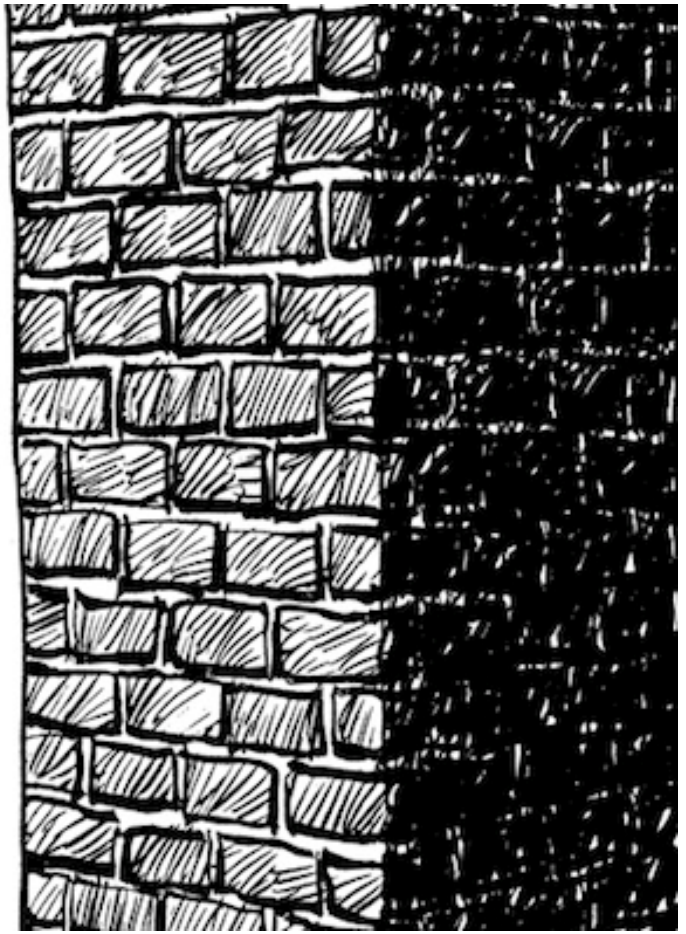
Get to know your code by instrumentation

Patrick Barel, Qualogy October 12, 2022

QUALOGY

Why?

Why?



How? DB Logging!

- Not available by default
  - DBMS_OUTPUT is NOT logging

My program

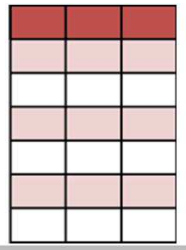
```
create or replace procedure my_program
is
begin
  dbms_output.put_line
   (q'[Start of my_program]');
  dbms_output.put_line
   (q'[Start doing important stuff]');
  -- do all the important stuff here
  -- write more logging if necessary
  dbms_output.put_line
    (q'[End doing important stuff]');
  dbms_output.put_line
    (q'[Start doing even more important stuff]');
  -- do even more important stuff here
  -- and write even more logging if necessary
  dbms_output.put_line
    (q'[End doing even more important stuff]');
  dbms_output.put_line
    (q'[End of my_program]');
end;
/
```

```
DEMO@demo> exec my_program

PL/SQL procedure successfully completed.

DEMO@demo> set serveroutput on size unlimited
DEMO@demo> exec my_program
Start of my_program
Start doing important stuff
End doing important stuff
Start doing even more important stuff
End doing even more important stuff
End of my_program

PL/SQL procedure successfully completed.

DEMO@demo>
```

How? DB Logging!

- Not available by default
  - DBMS_OUTPUT is NOT logging
- Build your own

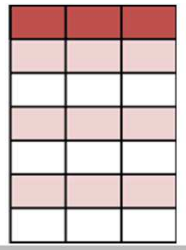
My program



```
create or replace procedure my_program
is
begin
  insert into t_log(info)
   values (q'[Start of my_program]');
  insert into t_log(info)
   values (q'[Start doing important stuff]');
  -- do all the important stuff here
  -- write more logging if necessary
  insert into t_log(info)
   values (q'[End doing important stuff]');
  insert into t_log(info)
   values (q'[Start doing even more important stuff]');
  -- do even more important stuff here
  -- and write even more logging if necessary
  insert into t_log(info)
   values (q'[End doing even more important stuff]');
  insert into t_log(info)
   values (q'[End of my_program]');
end;
/
```

```
DEMO@demo> create table t_log
( id number generated always as identity
, info varchar2(4000)
, time_stamp timestamp default systimestamp
)
/
Table created
DEMO@demo>exec my_program
PL/SQL procedure successfully completed
DEMO@demo>
```

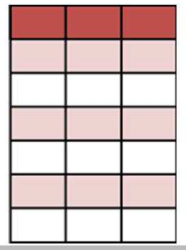
My program



```
create or replace procedure my_program
is
begin
  insert into t_log(info)
   values (q'[Start of my_program]');
  insert into t_log(info)
   values (q'[Start doing important stuff]');
  -- do all the important stuff here
  -- write more logging if necessary
  insert into t_log(info)
   values (q'[End doing important stuff]');
  insert into t_log(info)
   values (q'[Start doing even more important stuff]');
  -- do even more important stuff here
  -- and write even more logging if necessary
  insert into t_log(info)
   values (q'[End doing even more important stuff]');
  insert into t_log(info)
   values (q'[End of my_program]');
end;
/
```

```
DEMO@demo> set linesize 45
DEMO@demo> column id format 999
DEMO@demo> column info format a40
DEMO@demo> column time_stamp format a30
DEMO@demo> select *
   from t_log
 /
```

My program



```
create or replace procedure my_program
is
begin
  insert into t_log(info)
   values (q'[Start of my_program]');
  insert into t_log(info)
   values (q'[Start doing important stuff]');
  -- do all the important stuff here
  -- write more logging if necessary
  insert into t_log(info)
   values (q'[End doing important stuff]');
  insert into t_log(info)
   values (q'[Start doing even more important stuff]');
  -- do even more important stuff here
  -- and write even more logging if necessary
  insert into t_log(info)
   values (q'[End doing even more important stuff]');
  insert into t_log(info)
   values (q'[End of my_program]');
end;
/
```

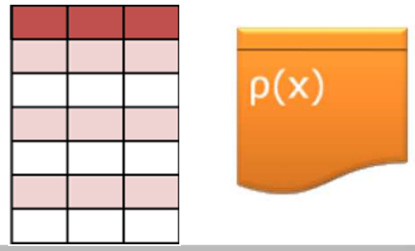
```
  ID INFO
---- ----------------------------------------
TIME_STAMP
------------------------------
   1 Start of my_program
25-FEB-18 05.38.51.242213 AM
   2 Start doing important stuff
25-FEB-18 05.38.51.255335 AM
   3 End doing important stuff
25-FEB-18 05.38.51.259890 AM
   4 Start doing even more important stuff
25-FEB-18 05.38.51.264286 AM
   5 End doing even more important stuff
25-FEB-18 05.38.51.269570 AM
   6 End of my_program
25-FEB-18 05.38.51.274809 AM

6 rows selected.
DEMO@demo> rollback
/
DEMO@demo> select *
  from t_log
/
```

My program

```
create or replace procedure my_program
is
begin
  insert into t_log(info)
   values (q'[Start of my_program]');
  insert into t_log(info)
   values (q'[Start doing important stuff]');
  -- do all the important stuff here
  -- write more logging if necessary
  insert into t_log(info)
   values (q'[End doing important stuff]');
  insert into t_log(info)
   values (q'[Start doing even more important stuff]');
  -- do even more important stuff here
  -- and write even more logging if necessary
  insert into t_log(info)
   values (q'[End doing even more important stuff]');
  insert into t_log(info)
   values (q'[End of my_program]');
end;
/
```

```
no rows selected.

DEMO@demo>
```

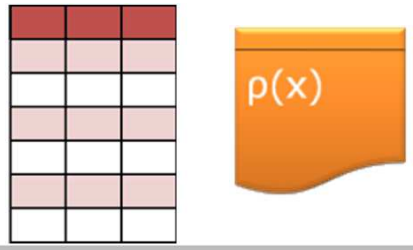
My program




```
create or replace procedure my_program
is
begin
  logthis(q'[Start of my program]');
  logthis(q'[Start doing important stuff]');
  insert into the_log(info) stuff here
  -- write more logging if necessary
  logthis(q'[Finished important stuff]');
  logthis(q'[Start doing even more important stuff]');
  insert into t_log(info) even more important stuff here
  -- and write even more logging if necessary
  logthis(q'[End doing even more important stuff]');
  logthis(q'[End of my program]');
end; do even more important stuff here
/ -- and write even more logging if necessary
  insert into t_log(info)
   values (q'[End doing even more important stuff]');
  insert into t_log(info)
   values (q'[End of my_program]');
end;
/
```

```
DEMO@demo> create or replace procedure
   logthis(logtext_in in varchar2)
 is

 begin
   insert into t_log(info) values (logtext_in);

 exception
   when others then

     raise;
 end logthis;
 /

Procedure created
DEMO@demo> exec my_program
 PL/SQL procedure successfully completed
DEMO@demo> select *
  from t_log
/
```

My program




```
create or replace procedure my_program
is
begin
  logthis(q'[Start of my_program]');
  logthis(q'[Start doing important stuff]');
  -- do all the important stuff here
  -- write more logging if necessary
  logthis(q'[End doing important stuff]');
  logthis(q'[Start doing even more important stuff]');
  -- do even more important stuff here
  -- and write even more logging if necessary
  logthis(q'[End doing even more important stuff]');
  logthis(q'[End of my_program]');
end;
/
```

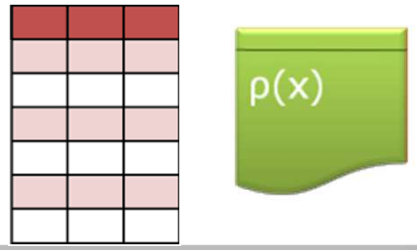
```
  ID INFO
---- ----------------------------------------
TIME_STAMP
------------------------------
   1 Start of my_program
25-FEB-18 05.38.51.242213 AM
   2 Start doing important stuff
25-FEB-18 05.38.51.255335 AM
   3 End doing important stuff
25-FEB-18 05.38.51.259890 AM
   4 Start doing even more important stuff
25-FEB-18 05.38.51.264286 AM
   5 End doing even more important stuff
25-FEB-18 05.38.51.269570 AM
   6 End of my_program
25-FEB-18 05.38.51.274809 AM

6 rows selected.
DEMO@demo> rollback
/
DEMO@demo> select *
  from t_log
/
```

My program

```
create or replace procedure my_program
is
begin
  logthis(q'[Start of my_program]');
  logthis(q'[Start doing important stuff]');
  -- do all the important stuff here
  -- write more logging if necessary
  logthis(q'[End doing important stuff]');
  logthis(q'[Start doing even more important stuff]');
  -- do even more important stuff here
  -- and write even more logging if necessary
  logthis(q'[End doing even more important stuff]');
  logthis(q'[End of my_program]');
end;
/
```

```
no rows selected.

DEMO@demo>
```

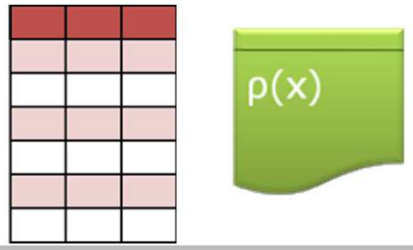
My program




```
create or replace procedure my_program
is
begin
  logthis(q'[Start of my_program]');
  logthis(q'[Start doing important stuff]');
  -- do all the important stuff here
  -- write more logging if necessary
  logthis(q'[End doing important stuff]');
  logthis(q'[Start doing even more important stuff]');
  -- do even more important stuff here
  -- and write even more logging if necessary
  logthis(q'[End doing even more important stuff]');
  logthis(q'[End of my_program]');
end;
/
```

```
DEMO@demo> create or replace procedure
   logthis(logtext_in in varchar2)
 is
   pragma autonomous_transaction;
 begin
   insert into t_log(info) values (logtext_in);
   commit;
 exception
   when others then
     rollback;
     raise;
 end logthis;
 /

Procedure created
DEMO@demo> exec my_program
 PL/SQL procedure successfully completed
DEMO@demo> select *
  from t_log
/
```

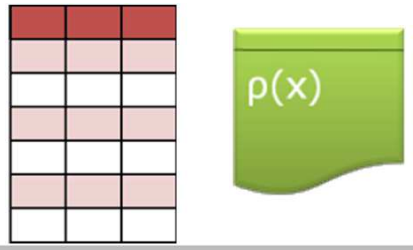
My program




```
create or replace procedure my_program
is
begin
  logthis(q'[Start of my_program]');
  logthis(q'[Start doing important stuff]');
  -- do all the important stuff here
  -- write more logging if necessary
  logthis(q'[End doing important stuff]');
  logthis(q'[Start doing even more important stuff]');
  -- do even more important stuff here
  -- and write even more logging if necessary
  logthis(q'[End doing even more important stuff]');
  logthis(q'[End of my_program]');
end;
/
```

```
  ID INFO
---- ----------------------------------------
TIME_STAMP
-----------------------------
   1 Start of my_program
25-FEB-18 05.38.51.242213 AM
   2 Start doing important stuff
25-FEB-18 05.38.51.255335 AM
   3 End doing important stuff
25-FEB-18 05.38.51.259890 AM
   4 Start doing even more important stuff
25-FEB-18 05.38.51.264286 AM
   5 End doing even more important stuff
25-FEB-18 05.38.51.269570 AM
   6 End of my_program
25-FEB-18 05.38.51.274809 AM

6 rows selected.
DEMO@demo> rollback
/
DEMO@demo> select *
  from t_log
/
```

My program




```
create or replace procedure my_program
is
begin
  logthis(q'[Start of my_program]');
  logthis(q'[Start doing important stuff]');
  -- do all the important stuff here
  -- write more logging if necessary
  logthis(q'[End doing important stuff]');
  logthis(q'[Start doing even more important stuff]');
  -- do even more important stuff here
  -- and write even more logging if necessary
  logthis(q'[End doing even more important stuff]');
  logthis(q'[End of my_program]');
end;
/
```

```
  ID INFO
---- ----------------------------------------
TIME_STAMP
------------------------------
   1 Start of my_program
25-FEB-18 05.38.51.242213 AM
   2 Start doing important stuff
25-FEB-18 05.38.51.255335 AM
   3 End doing important stuff
25-FEB-18 05.38.51.259890 AM
   4 Start doing even more important stuff
25-FEB-18 05.38.51.264286 AM
   5 End doing even more important stuff
25-FEB-18 05.38.51.269570 AM
   6 End of my_program
25-FEB-18 05.38.51.274809 AM

6 rows selected.
DEMO@demo>
```

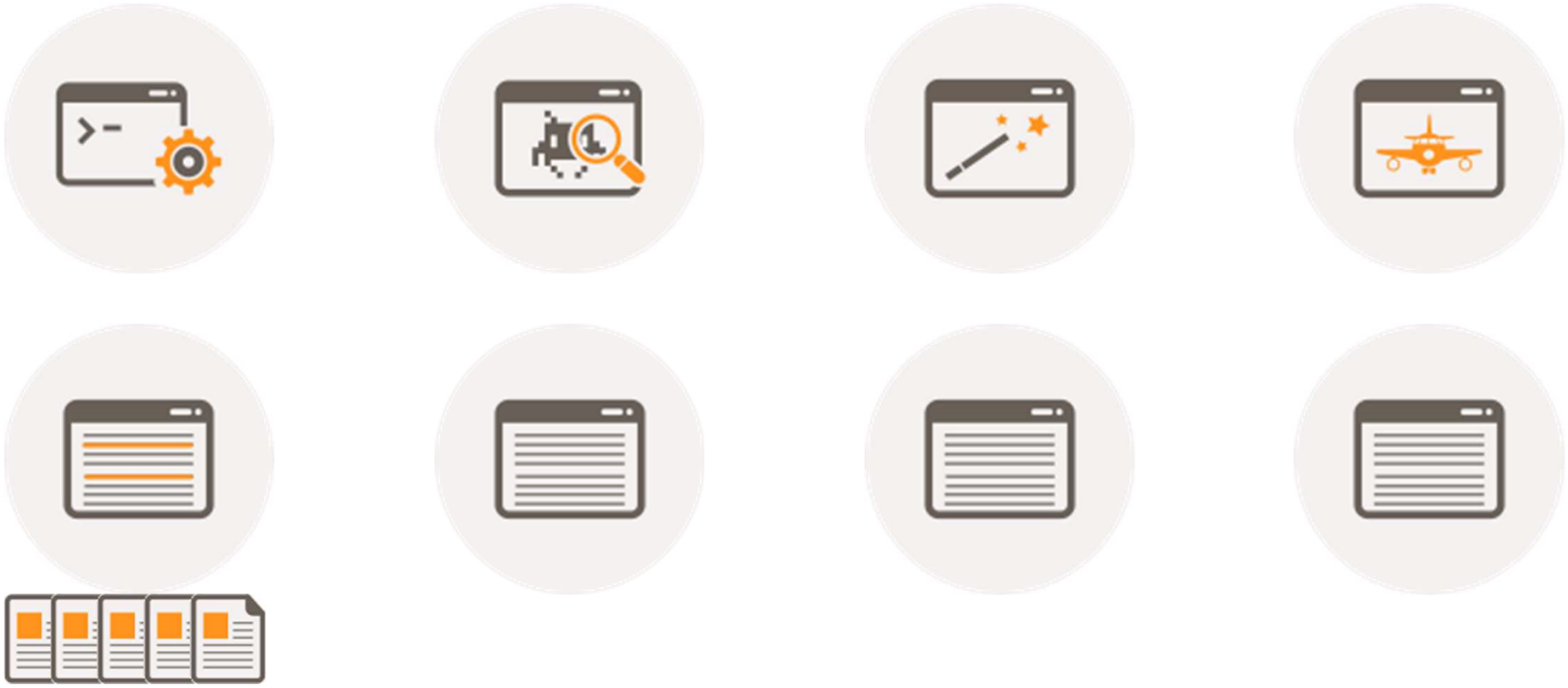
How? DB Logging!

- Not available by default
  - DBMS_OUTPUT is NOT logging
- Build your own
- Use Open Source

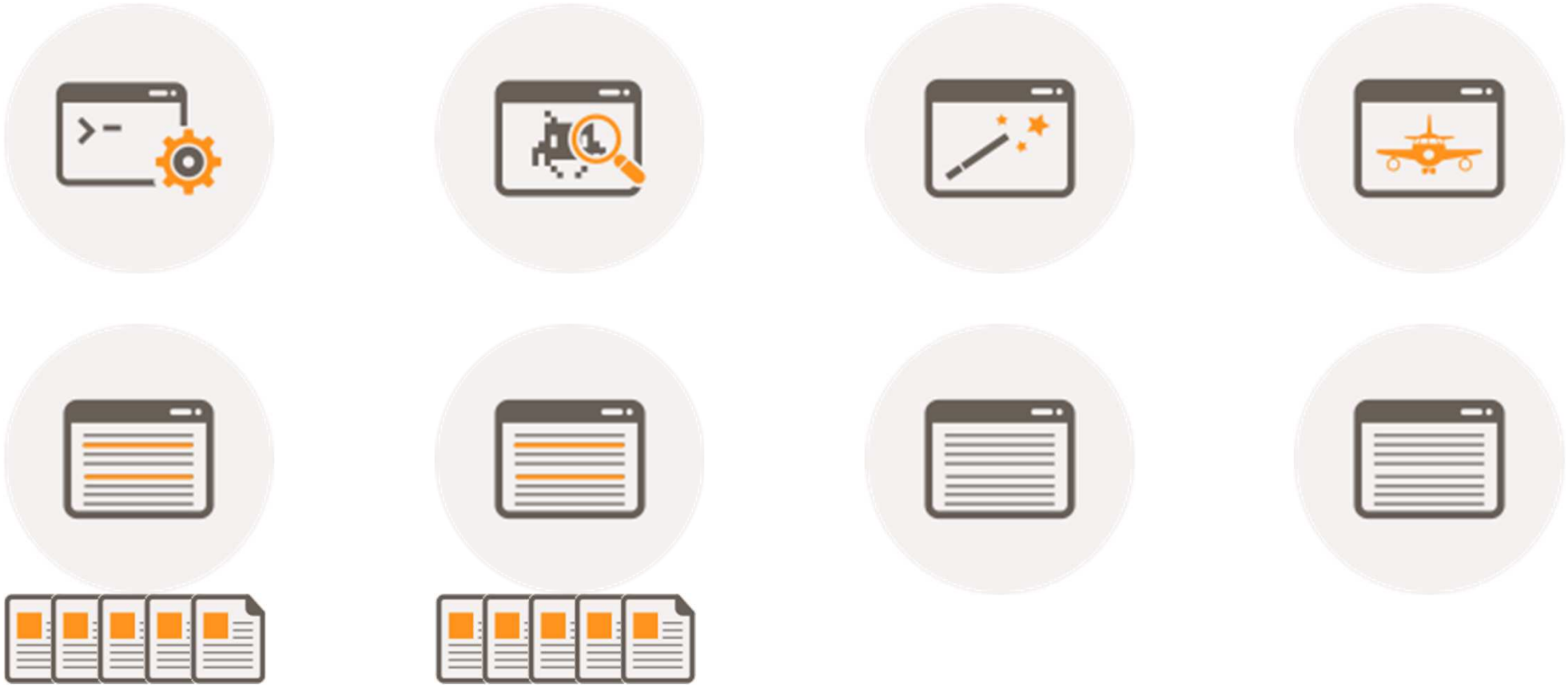
How? DB Logging!

- Not available by default
  - DBMS_OUTPUT is NOT logging
- Build your own
- Use Open Source – with adaptions

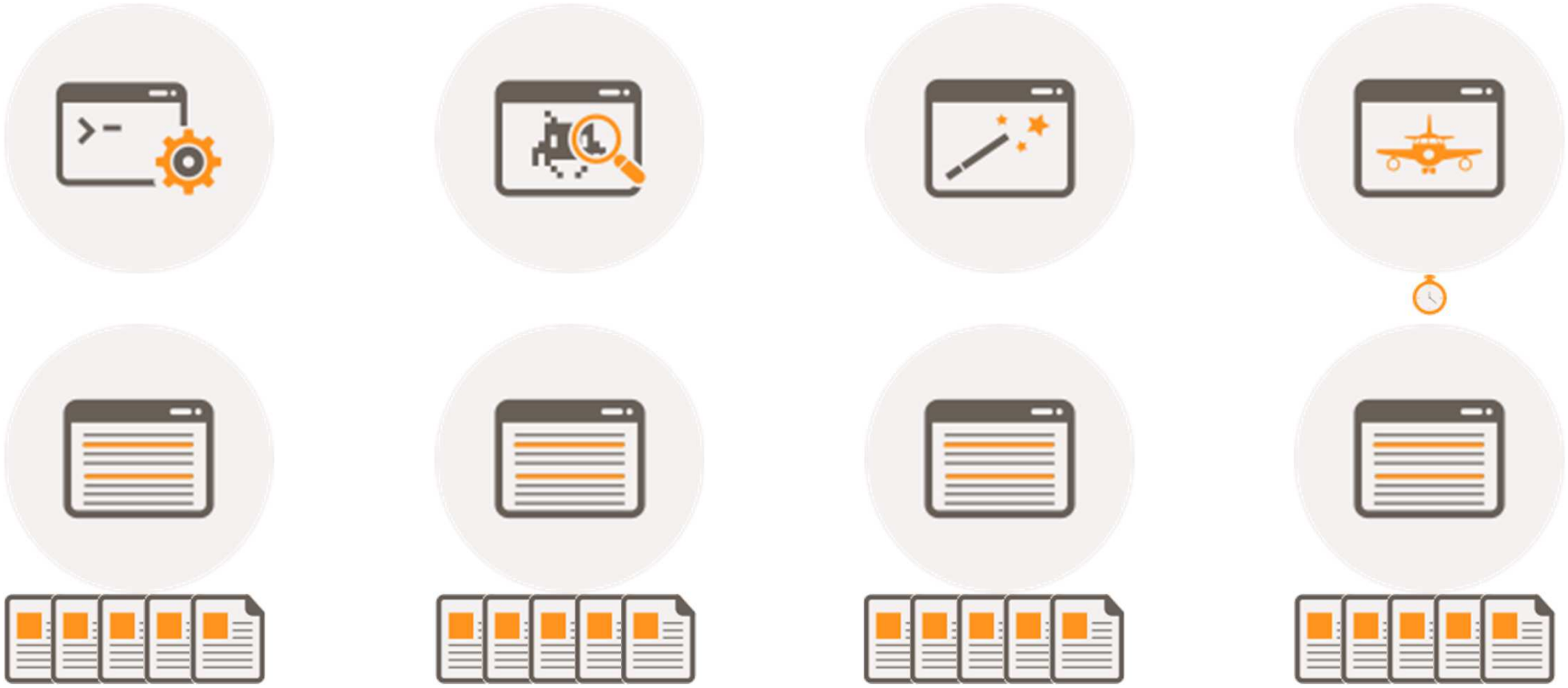
Implementing logging



Implementing logging



Implementing logging



How?

Logger

- Install in a separate schema
- Grant access to individual users or public
- Cleanup jobs are installed automatically.
  - Cleanup logger information older than x days
    Severity lower than threshold
  - ‘expired’ preferences

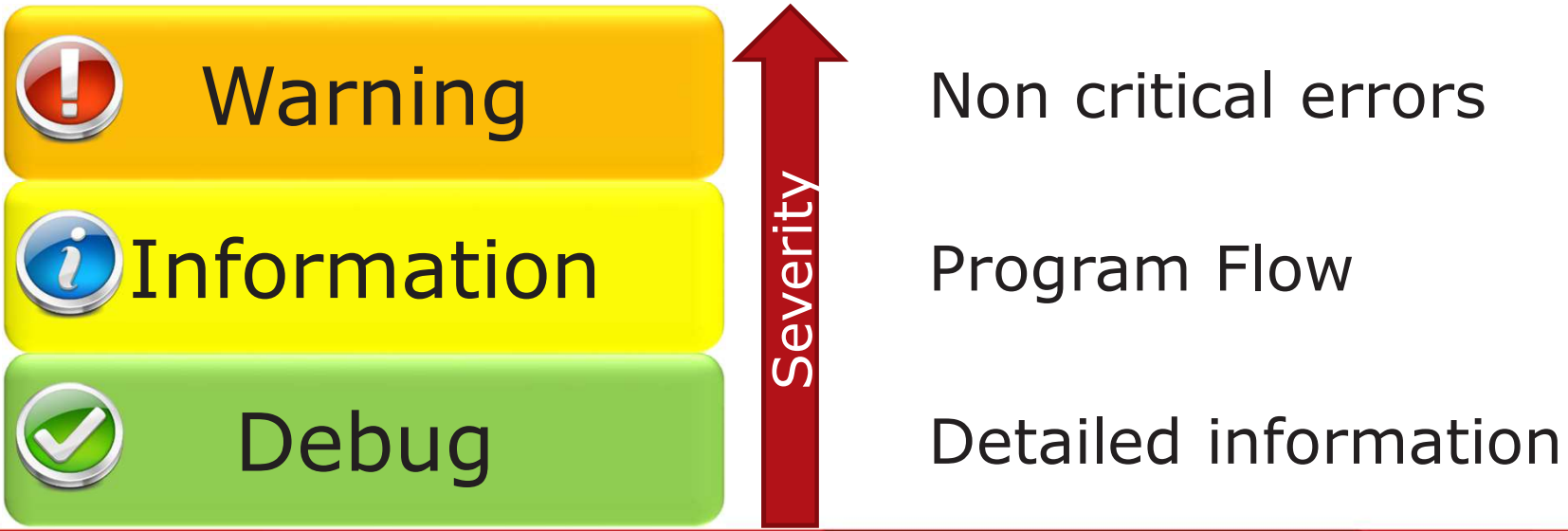
Log Levels

Detailed information

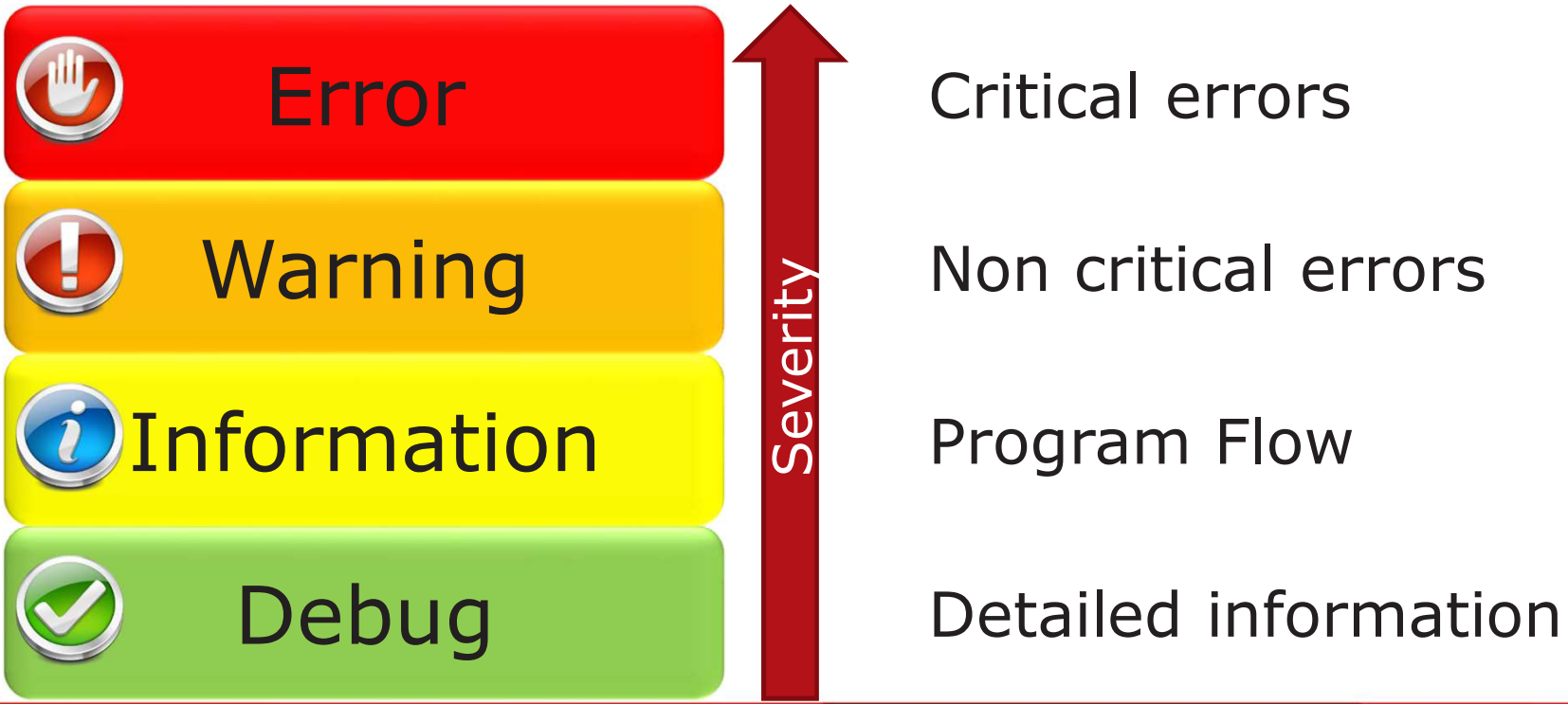
Log Levels


Information
Debug
Severity
Program Flow
Detailed information

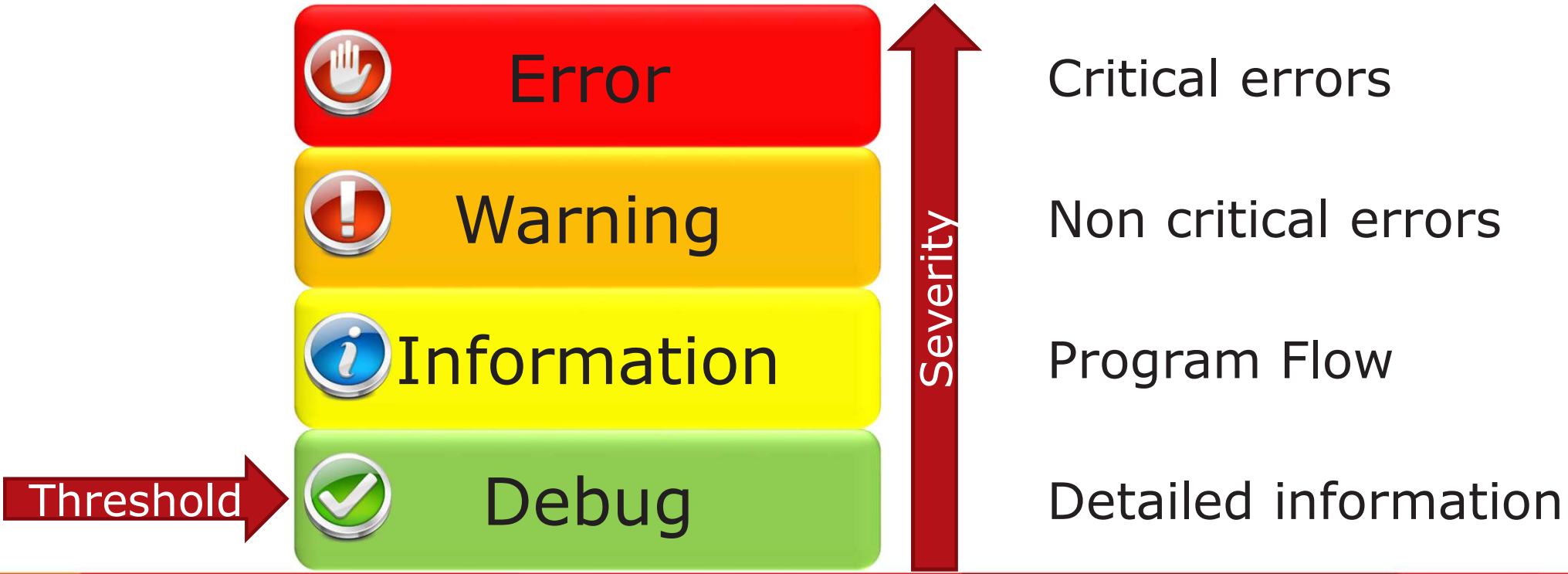
Log Levels


Warning
Information
Debug
Severity
Non critical errors
Program Flow
Detailed information

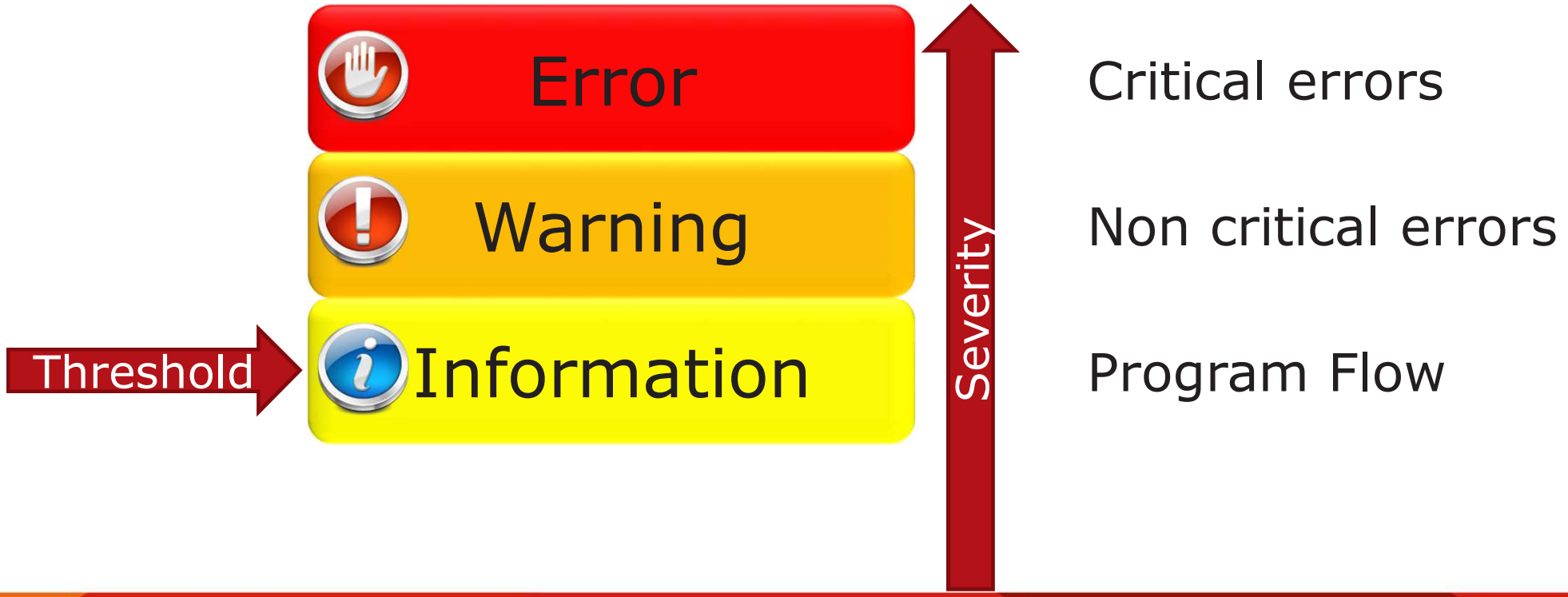
Log Levels


Error
Warning
Information
Debug
Severity
Critical errors
Non critical errors
Program Flow
Detailed information

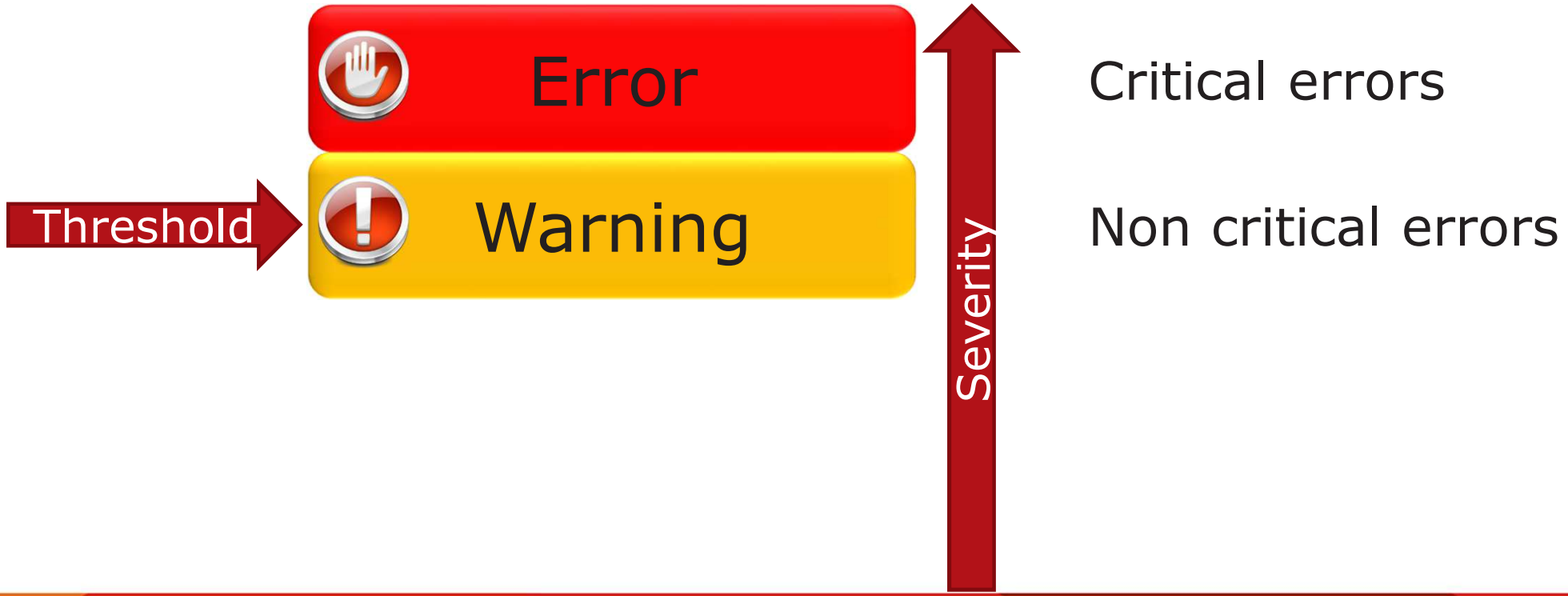
Log Levels


Error
Warning
Information
Debug
Threshold
Severity
Critical errors
Non critical errors
Program Flow
Detailed information

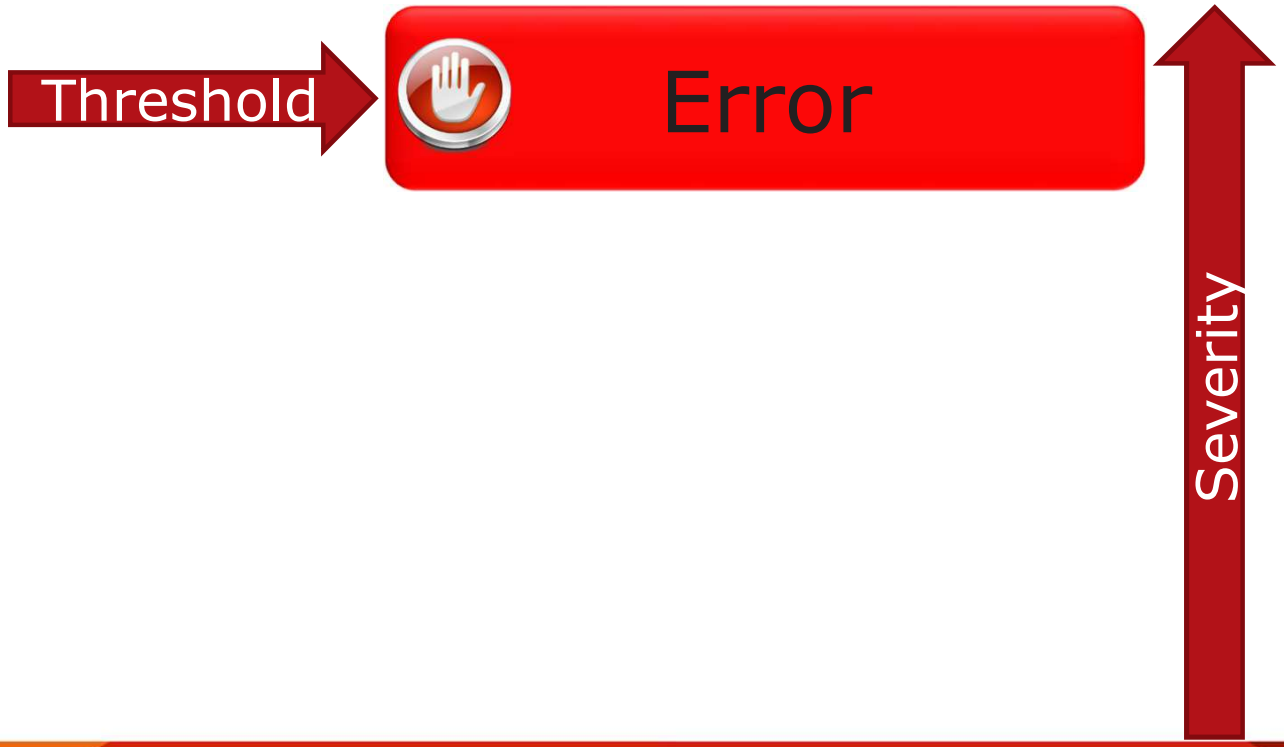
Log Levels


Error
Warning
Information
Threshold
Severity
Critical errors
Non critical errors
Program Flow

Log Levels


Error
Warning
Threshold
Severity
Critical errors
Non critical errors

Log Levels



Critical errors

Current Loglevel – One switch to rule them all

- Error – All critical errors

Current Loglevel – One switch to rule them all

- Error – All critical errors
- Warning – All non critical errors

Current Loglevel – One switch to rule them all

- Error – All critical errors
- Warning – All non critical errors
- Information – Information for instance about program flow

Current Loglevel – One switch to rule them all

- Error – All critical errors
- Warning – All non critical errors
- Information – Information for instance about program flow
- Debug – All information you might be interested in


HOW
TO...

How to use

```
create or replace package body logger_example is
  gc_scope_prefix constant varchar2(258) := lower(user) || '.' ||
                                            lower($$plsql_unit) || '.';

  procedure foo is ...

  function bar return number is ...
end logger_example;
```

Objectnames can be 128 since 12.2

How to use

```
create or replace package body logger_example is
  gc_scope_prefix constant varchar2(258) := lower(user) || '.' ||
                                            lower($$plsql_unit) || '.';

  procedure foo is
    l_scope  logger_logs.scope%type := gc_scope_prefix || 'foo';
    l_params logger.tab_param;
    l        number;
  begin
    ...
  end foo;

  function bar return number is ...
end logger_example;
```

How to use

```
create or replace package body logger_example is
  gc_scope_prefix constant varchar2(258) := lower(user) || '.' ||
                                            lower($$plsql_unit) || '.';

  procedure foo is
    l_scope  logger_logs.scope%type := gc_scope_prefix || 'foo';
    l_params logger.tab_param;
    l        number;
  begin
    logger.log_information(p_text  => '(' || $$plsql_line || ') Start'
                          ,p_scope => l_scope);
    ...
  end foo;

  function bar return number is ...
end logger_example;
```

How to use

```
create or replace package body logger_example is
  gc_scope_prefix constant varchar2(258) := lower(user) || '.' ||
                                            lower($$plsql_unit) || '.';

  procedure foo is
    l_scope  logger_logs.scope%type := gc_scope_prefix || 'foo';
    l_params logger.tab_param;
    l        number;
  begin
    logger.log_information(p_text  => '(' || $$plsql_line || ') Start'
                          ,p_scope => l_scope);
    logger.log(p_text  => '(' || $$plsql_line || ') ' ||
                          'Do something important'
              ,p_scope => l_scope);
    ...
  end foo;

  function bar return number is ...
end logger_example;
```

How to use

```
create or replace package body logger_example is
  gc_scope_prefix constant varchar2(258) := lower(user) || '.' ||
                                            lower($$plsql_unit) || '.';

  procedure foo is
    l_scope  logger_logs.scope%type := gc_scope_prefix || 'foo';
    l_params logger.tab_param;
    l        number;
  begin
    logger.log_information(p_text  => '(' || $$plsql_line || ') Start'
                          ,p_scope => l_scope);
    logger.log(p_text  => '(' || $$plsql_line || ') ' ||
                          'Do something important'
              ,p_scope => l_scope);
    for indx in 1 .. 10 loop
      l := indx * indx;
      if l > 100 then
        logger.log_warning(p_text  => '(' || $$plsql_line || ') ' || '[' ||
                                      sqlcode || ']' || sqlerrm ||
                                      'Value higher than expected'
                          ,p_scope => l_scope
                          ,p_extra => dbms_utility.format_error_backtrace);
      end if;
```

How to use

```
procedure foo is
  l_scope  logger_logs.scope%type := gc_scope_prefix || 'foo';
  l_params logger.tab_param;
  l        number;
begin
  logger.log_information(p_text  => '(' || $$plsql_line || ') Start'
                         ,p_scope => l_scope);
  logger.log(p_text  => '(' || $$plsql_line || ') ' ||
                        'Do something important'
            ,p_scope => l_scope);
  for indx in 1 .. 10 loop
    l := indx * indx;
    if l > 100 then
      logger.log_warning(p_text  => '(' || $$plsql_line || ') ' || '[' ||
                                    sqlcode || ']' || sqlerrm ||
                                    'Value higher than expected'
                         ,p_scope => l_scope
                         ,p_extra => dbms_utility.format_error_backtrace);
    end if;
  end loop;
  logger.log_information(p_text  => '(' || $$plsql_line || ') End'
                         ,p_scope => l_scope);
exception
```

How to use

```
begin
  logger.log_information(p_text  => '(' || $$plsql_line || ') Start'
                        ,p_scope => l_scope);
  logger.log(p_text  => '(' || $$plsql_line || ') ' ||
                        'Do something important'
            ,p_scope => l_scope);
  for indx in 1 .. 10 loop
    l := indx * indx;
    if l > 100 then
      logger.log_warning(p_text  => '(' || $$plsql_line || ') ' || '[' ||
                                    sqlcode || ']' || sqlerrm ||
                                    'Value higher than expected'
                        ,p_scope => l_scope
                        ,p_extra => dbms_utility.format_error_backtrace);
    end if;
  end loop;
  logger.log_information(p_text  => '(' || $$plsql_line || ') End'
                        ,p_scope => l_scope);
exception
  when others then
    logger.log_error(p_text  => '(' || $$plsql_line || ') ' || '[' ||
                                sqlcode || ']' || sqlerrm ||
                                'Something bad happened'
```

How to use

```
                              'Do something important'
                ,p_scope => l_scope);
    for indx in 1 .. 10 loop
      l := indx * indx;
      if l > 100 then
        logger.log_warning(p_text  => '(' || $$plsql_line || ') ' || '[' ||
                                      sqlcode || ']' || sqlerrm ||
                                      'Value higher than expected'
                           ,p_scope => l_scope
                           ,p_extra => dbms_utility.format_error_backtrace);
      end if;
    end loop;
    logger.log_information(p_text  => '(' || $$plsql_line || ') End'
                           ,p_scope => l_scope);
  exception
    when others then
      logger.log_error(p_text  => '(' || $$plsql_line || ') ' || '[' ||
                                  sqlcode || ']' || sqlerrm ||
                                  'Something bad happened'
                       ,p_scope => l_scope
                       ,p_extra => dbms_utility.format_error_backtrace);
  end foo;

  function bar return number is
```

How to use

```
function bar return number is
  l_scope  logger_logs.scope%type := gc_scope_prefix || 'bar';
  l_params logger.tab_param;
  l        number;
begin
  logger.log_information(p_text  => '(' || $$plsql_line || ') Start'
                        ,p_scope => l_scope);
  logger.log(p_text  => '(' || $$plsql_line || ') ' ||
                        'Do something important'
            ,p_scope => l_scope);
  for indx in 1 .. 10 loop
    l := indx * indx;
    if l > 100 then
      logger.log_warning(p_text  => '(' || $$plsql_line || ') ' || '[' ||
                                    sqlcode || ']' || sqlerrm ||
                                    'Value higher than expected'
                        ,p_scope => l_scope
                        ,p_extra => dbms_utility.format_error_backtrace);
    end if;
  end loop;
  logger.log_information(p_text  => '(' || $$plsql_line || ') End'
                        ,p_scope => l_scope);
  return l;
exception
```


11
12

Performance impact

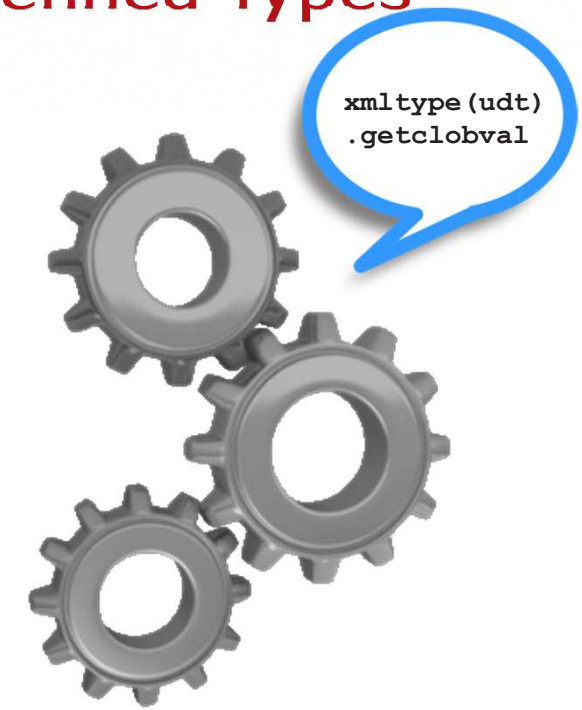
	DEBUG	INFORMATION	WARNING	ERROR	OFF
NO LOGGING	1349.898	1366.551	1388.812	1346.186	1365.701
CONDITIONAL LOGGING	6753.144	3680.012	1406.855	1466.653	1442.098
WITH LOGGING	5016.108	2894.907	1463.405	1510.79	1495.589
NO LOGGING UDT	1413.5	1346.872	1350.788	1358.918	1366.044
CONDITIONAL LOG UDT	7937.094	3510.1	1421.966	1452.766	1446.308
WITH LOGGING UDT	6346.311	4130.85	2258.73	2322.99	2348.085

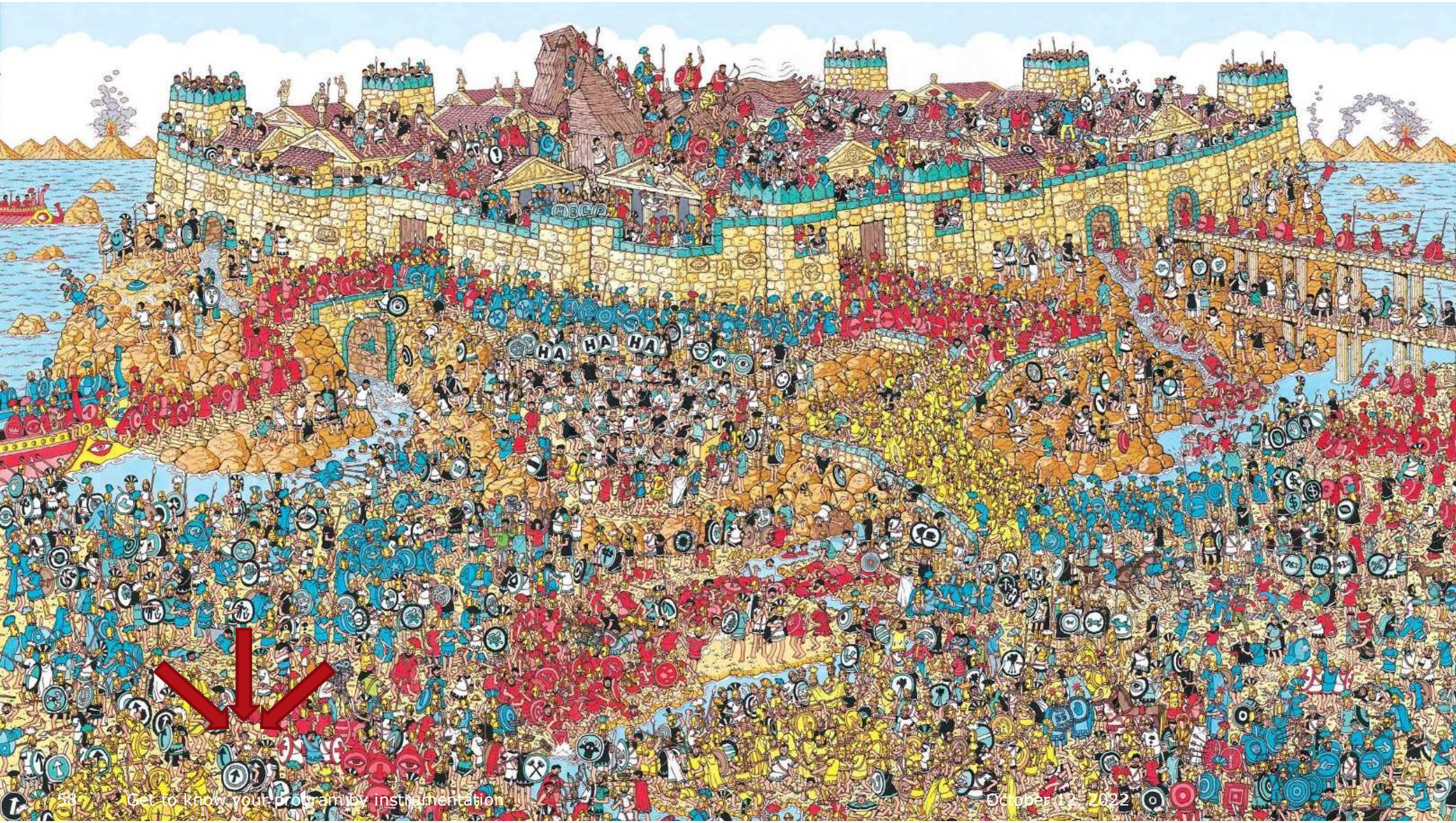
Times in milliseconds

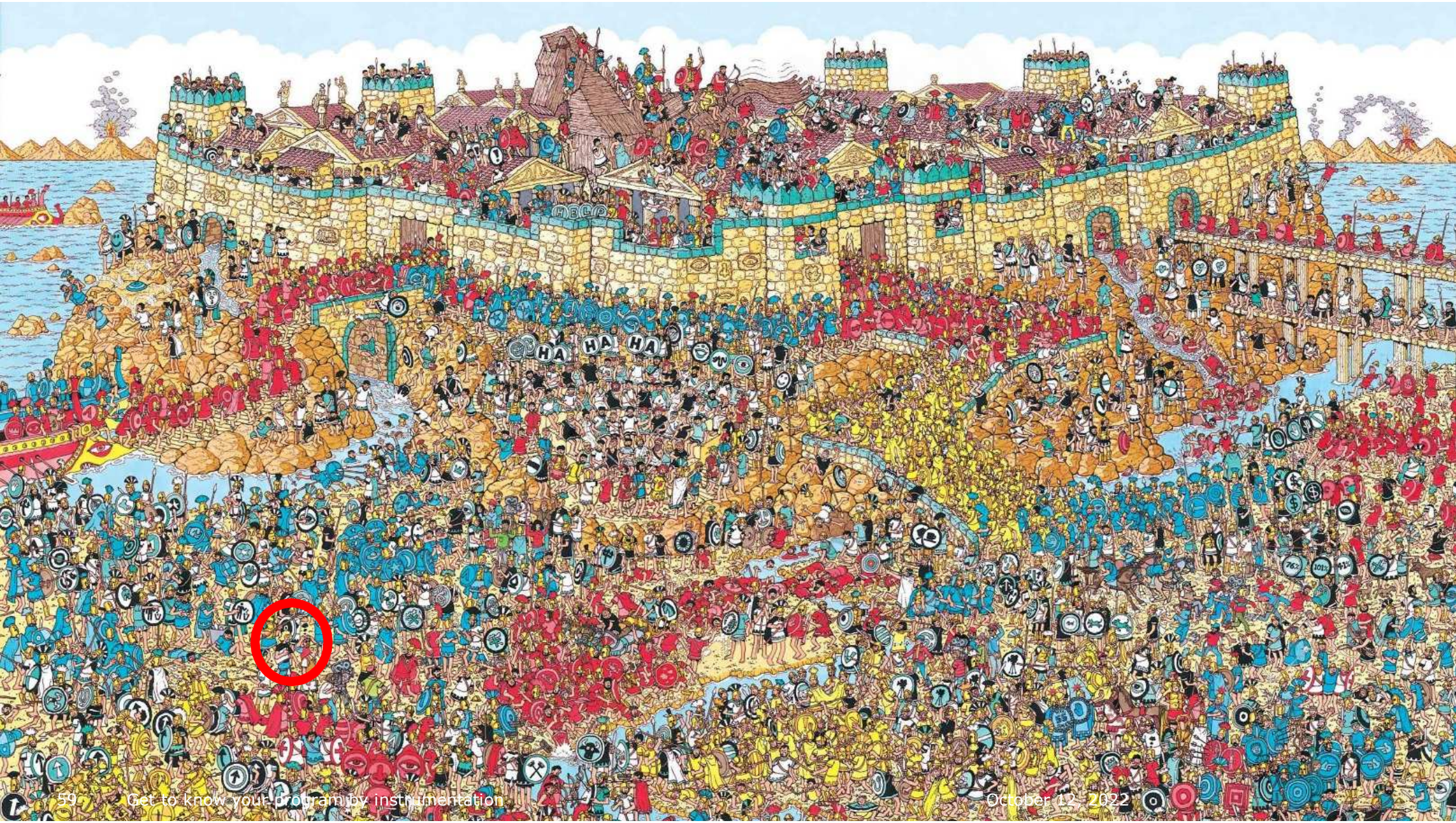
Repeat each test: 100
Repeat all tests: 20

Conversion of User Defined Types

Element	Type	
DEPTNO	NUMBER(2)	
DNAME	VARCHAR2(14)	
LOC	VARCHAR2(13)	
EMPS	**Element**	**Type**
	EMPNO	NUMBER(4)
	ENAME	VARCHAR2(10)
	JOB	VARCHAR2(9)
	MGR	NUMBER(4)
	HIREDATE	DATE
	SAL	NUMBER(7,2)
	COMM	NUMBER(7,2)
	DEPTNO	NUMBER(2)




HELP
HA HA HA


HELP
HA HA HA

How to find your lines of interest

```
select *
  from logger_logs ll
order by ll.id desc
```

	ID	LOGGER_LEVEL	TEXT	TIME_STAMP	SCOPE	MODULE
1	484641	8	(109) End	29-APR-18 05.07.47.340845 AM	demo.logger_example.showall	PL/SQL Develop
2	484640	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	29-APR-18 05.07.47.340333 AM	demo.logger_example.showall	PL/SQL Develop
3	484639	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	29-APR-18 05.07.47.340058 AM	demo.logger_example.showall	PL/SQL Develop
4	484638	16	(85) l = 225	29-APR-18 05.07.47.339739 AM	demo.logger_example.showall	PL/SQL Develop
5	484637	16	(83) indx = 15	29-APR-18 05.07.47.339345 AM	demo.logger_example.showall	PL/SQL Develop

How to find your lines of interest
Use subset of the columns

```
select ll.id, ll.sid, ll.logger_level, ll.text, ll.scope, ll.extra
  from logger_logs ll
 order by ll.id desc;
```

	ID	SID	LOGGER_LEVEL	TEXT	SCOPE	EXTRA
1	484641	156	8	(109) End	demo.logger_example.showall	<CLOB>
2	484640	156	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	demo.logger_example.showall	<CLOB>
3	484639	156	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	demo.logger_example.showall	<CLOB>
4	484638	156	16	(85) l = 225	demo.logger_example.showall	<CLOB>
5	484637	156	16	(83) indx = 15	demo.logger_example.showall	<CLOB>
6	484636	156	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	demo.logger_example.showall	<CLOB>
7	484635	156	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	demo.logger_example.showall	<CLOB>

How to find your lines of interest
Use SID

```
select ll.id, ll.sid, ll.logger_level, ll.text, ll.scope, ll.extra
  from logger_logs ll
 where ll.sid = &SID
 order by ll.id desc;
```

	ID	SID	LOGGER_LEVEL	TEXT	SCOPE	EXTRA
1	484641	156	8	(109) End	demo.logger_example.showall	<CLOB>
2	484640	156	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	demo.logger_example.showall	<CLOB>
3	484639	156	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	demo.logger_example.showall	<CLOB>
4	484638	156	16	(85) l = 225	demo.logger_example.showall	<CLOB>
5	484637	156	16	(83) indx = 15	demo.logger_example.showall	<CLOB>
6	484636	156	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	demo.logger_example.showall	<CLOB>
7	484635	156	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	demo.logger_example.showall	<CLOB>

How to find your lines of interest
Use LOGGER_LEVEL

```
select ll.id, ll.sid, ll.logger_level, ll.text, ll.scope, ll.extra
  from logger_logs ll
 where ll.logger_level = &LOGGER_LEVEL
 order by ll.id desc;
```

	ID	SID	LOGGER_LEVEL	TEXT	SCOPE	EXTRA
1	484641	156	8	(109) End	demo.logger_example.showall	<CLOB>
2	484640	156	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	demo.logger_example.showall	<CLOB>
3	484639	156	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	demo.logger_example.showall	<CLOB>
4	484638	156	16	(85) l = 225	demo.logger_example.showall	<CLOB>
5	484637	156	16	(83) indx = 15	demo.logger_example.showall	<CLOB>
6	484636	156	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	demo.logger_example.showall	<CLOB>
7	484635	156	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	demo.logger_example.showall	<CLOB>

How to find your lines of interest
Use TEXT

```
select ll.id, ll.sid, ll.logger_level, ll.text, ll.scope, ll.extra
  from logger_logs ll
 where ll.text like '%&TEXT%'
 order by ll.id desc;
```

	ID	SID	LOGGER_LEVEL	TEXT	SCOPE	EXTRA
1	484641	156	8	(109) End	demo.logger_example.showall	<CLOB>
2	484640	156	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	demo.logger_example.showall	<CLOB>
3	484639	156	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	demo.logger_example.showall	<CLOB>
4	484638	156	16	(85) l = 225	demo.logger_example.showall	<CLOB>
5	484637	156	16	(83) indx = 15	demo.logger_example.showall	<CLOB>
6	484636	156	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	demo.logger_example.showall	<CLOB>
7	484635	156	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	demo.logger_example.showall	<CLOB>

How to find your lines of interest
Use SCOPE

```
select ll.id, ll.sid, ll.logger_level, ll.text, ll.scope, ll.extra
  from logger_logs ll
 where ll.scope like '%&SCOPE%'
 order by ll.id desc;
```

	ID	SID	LOGGER_LEVEL	TEXT	SCOPE	EXTRA
1	484641	156	8	(109) End	demo.logger_example.showall	<CLOB>
2	484640	156	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	demo.logger_example.showall	<CLOB>
3	484639	156	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	demo.logger_example.showall	<CLOB>
4	484638	156	16	(85) l = 225	demo.logger_example.showall	<CLOB>
5	484637	156	16	(83) indx = 15	demo.logger_example.showall	<CLOB>
6	484636	156	2	(101) [0]ORA-0000: normal, successful completionValue is TOO high	demo.logger_example.showall	<CLOB>
7	484635	156	4	(87) [0]ORA-0000: normal, successful completionValue higher than expected	demo.logger_example.showall	<CLOB>

How to find your lines of interest
Use this script

```
/***
 *     ________               .__
 *     \_____  \  __ _______  |  |   ____   ____ ___.__.
 *      /  / \  \|  |  \__  \ |  |  /  _ \ / ___<   |  |
 *     /   \_/.  \  |  // __ \|  |_(  <_> ) /_/  >___  |
 *     \_____\ \_/____/(____  /____/\____/\___  // ____|
 *            \__>          \/           /_____/ \/
 */
/*
 * Script: CheckLogs
 * Author: Patrick Barel
 * Purpose: Retrieve the lines from the LOGGER_LOGS table based on the parameters
 *          provided
 * Parameters: - LastMinutes      : The number of minutes to look back in the table.
 *                                  No value means 45 days
 *             - TextLike         : This text should be in the TEXT field
 *             - TextNOTLike      : This text should NOT be in the TEXT field
 *             - ScopeLike        : The SCOPE should be like this
 *             - ScopeNOTLike     : The SCOPE should NOT be like this
 *             - MinLogLevel      : The minimum LOGGER_LEVEL you are interested in
```

How to find your lines of interest
Use this script

```
/*
 * Script: CheckLogs
 * Author: Patrick Barel
 * Purpose: Retrieve the lines from the LOGGER_LOGS table based on the parameters
 *          provided
 * Parameters: - LastMinutes     : The number of minutes to look back in the table.
 *                                 No value means 45 days
 *             - TextLike        : This text should be in the TEXT field
 *             - TextNOTLike     : This text should NOT be in the TEXT field
 *             - ScopeLike       : The SCOPE should be like this
 *             - ScopeNOTLike    : The SCOPE should NOT be like this
 *             - MinLogLevel     : The minimum LOGGER_LEVEL you are interested in
 *             - MaxLogLevel     : The maximum LOGGER_LEVEL you are interested in
 *             - UsernameLike    : The Username should be like this
 *             - UsernameNOTLike : The Username should NOT be like this
 * 20210208 PBA: Added log prefix and suffix to display the extra info, even
 *               when the level requested is lower
 * 20220308 PBA: Added User
 */
with theparameters as
```

How to find your lines of interest
Use this script

```
with theparameters as
 (select /* LastMinutes is given in minutes, this must be separated into days,
             hours and minutes */
         floor(nvl2('&LastMinutes', '&LastMinutes', '65536') / (24 * 60)) days
       , mod(floor(nvl2('&LastMinutes', '&LastMinutes', '65536') / 60), 24) hours
       , mod(nvl2('&LastMinutes', '&LastMinutes', '65536'), 60) minutes
         /* TextLike sent in must be prefixed and suffixed by a % */
       , '%' || '&TextLike' || '%' textlike
         /* if input for TextNOTLike is null then return some gibberish */
       , nvl2('&TextNOTLike', '%&TextNOTLike%', '%#^@*()%') textnotlike
         /* ScopeLike sent in must be prefixed and suffixed by a % */
       , '%' || '&ScopeLike' || '%' scopelike
         /* if input for ScopeNOTLike is null then return some gibberish */
       , nvl2('&ScopeNOTLike', '%&ScopeNOTLike%', '%#^@*()%') scopenotlike
         /* If no value is provided use a ridiculously low value */
       , nvl2('&MinLogLevel', '&MinLogLevel', '-65536') minloglevel
         /* If no value is provided use a ridiculously high value */
       , nvl2('&MaxLogLevel', '&MaxLogLevel', '65536') maxloglevel
         /* UsernameLike sent in must be prefixed and suffixed by a % */
       , '%' || '&UsernameLike' || '%' usernamelike
```

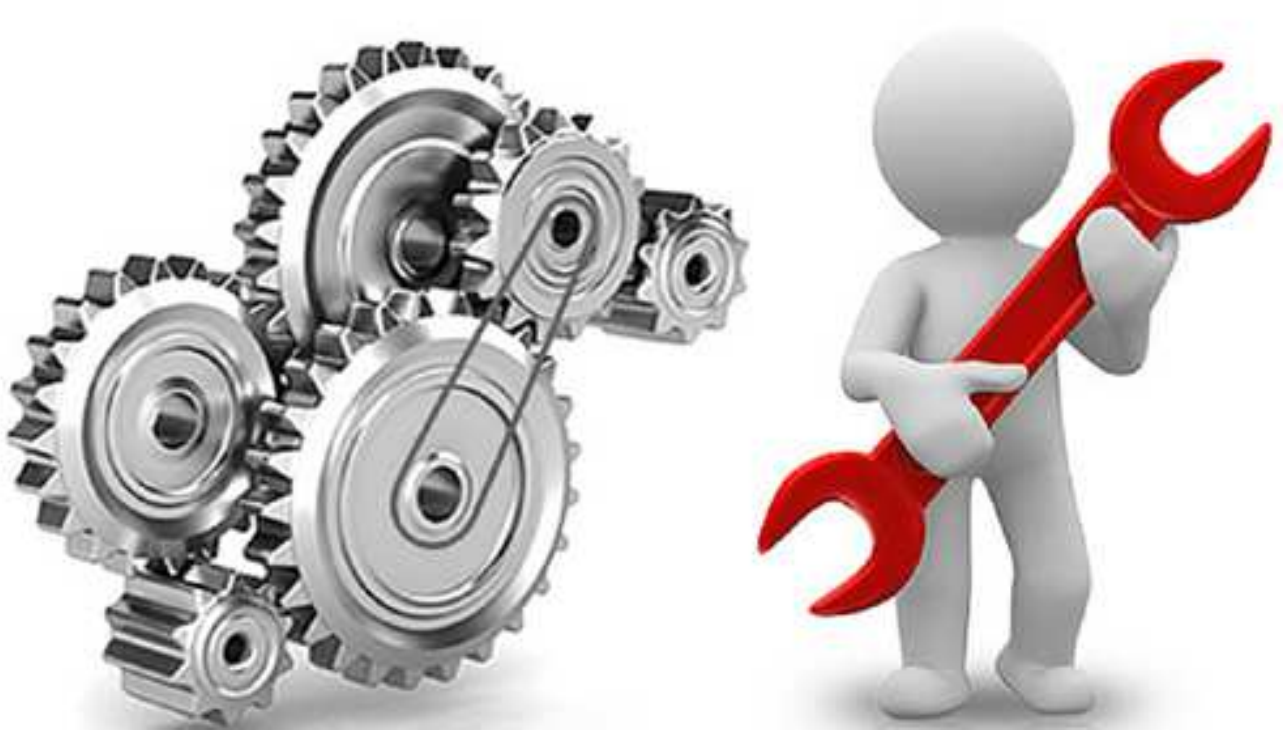
How to find your lines of interest
Use this script

```
select ll.id
     , ll.sid
     , ll.logger_level
     , ll.time_stamp
     , ll.scope
     , ll.text
     , ll.extra
     , ll.user_name
     , case
         when ll.text not like '%Start%'
           then ll.time_stamp - lead(ll.time_stamp) over
             (partition by ll.sid, ll.scope order by ll.time_stamp desc)
         else null
       end interval
     , systimestamp
     , aa_prefix
     , aa_suffix
  from logger_logs ll
 cross join theparameters
 where 1 = 1
```

How to find your lines of interest
Use this script

```
cross join theparameters
where 1 = 1
  and ll.time_stamp   >        systimestamp - to_dsinterval(theparameters.days
                                            || ' ' || theparameters.hours
                                            || ':' || theparameters.minutes
                                            || ':00')
  and ll.text             like theparameters.textlike
  and ll.text         not like theparameters.textnotlike
  and ll.scope            like theparameters.scopelike
  and ll.scope        not like theparameters.scopenotlike
  and ll.user_name        like theparameters.usernamelike
  and ll.user_name    not like theparameters.usernamenotlike
  and ll.logger_level >=       theparameters.minloglevel
  and (   ll.logger_level <=       theparameters.maxloglevel
       or (    theparameters.maxloglevel >= 2
           and ll.scope like aa_prefix || '%' || aa_suffix || '%'
          )
      )
order by ll.time_stamp desc
/
```

Enhancements



Current Loglevel – One switch to rule them all

Log
Level

New Loglevel – One switch per scope

Master
Level

New Loglevel – One switch per scope

Master Level

Current Development

New Loglevel – One switch per scope

Master Level

Current Development

Read Settings

New Loglevel – One switch per scope

Master Level

Current Development

Read Settings

Write log information

New Loglevel – One switch per scope

Master Level

Current Development

Read Settings

Write log information

…

…

…

Test program

```
create or replace package logger_specific as
  procedure normal_behaviour;
  procedure just_show_errors;
  procedure this_needs_investigation;
  procedure run_all;
end;
/
create or replace package body logger_specific as
  ...
  procedure normal_behaviour is
  ...
  procedure just_show_errors is
  ...
  procedure this_needs_investigation as
  ...
 procedure run_all is
  ...
    logger_specific.normal_behaviour;
    logger_specific.just_show_errors;
    logger_specific.this_needs_investigation;
  ...
  end;
end;
/
```

Same content

```
procedure normal_behaviour          is
   l_scope  logger_logs.scope%type := gc_scope_prefix || 'normal_behaviour';
   l_params logger.tab_param;
 begin
   logger.log_information(p_text => '(' || $$plsql_line || ') Start', p_scope => l_scope);
   -- write a debug message
   logger.log(p_text => '('|| $$plsql_line || ') ' || 'This is a debug message in '||l_scope
             ,p_scope => l_scope);
   -- write an information message
   logger.log_information(p_text => '(' || $$plsql_line || ') ' || 'This is an information message in '||l_scope
                         ,p_scope => l_scope);
   -- write a warning message
   logger.log_warning(p_text  => '(' || $$plsql_line || ') ' || '[' || sqlcode || ']' || sqlerrm ||
                                 'This is a warning message in '||l_scope
                     ,p_scope => l_scope
                     ,p_extra => dbms_utility.format_error_backtrace);
   -- write an error message
   logger.log_error(p_text  => '(' || $$plsql_line || ') ' || '[' || sqlcode || ']' || sqlerrm ||
                               'This is an error message in '||l_scope
                   ,p_scope => l_scope
                   ,p_extra => dbms_utility.format_error_backtrace);
   logger.log_information(p_text => '(' || $$plsql_line || ') End', p_scope => l_scope);
 end;
```

Same content

```
procedure just_show_errors         is
   l_scope  logger_logs.scope%type := gc_scope_prefix || 'just_show_errors';
   l_params logger.tab_param;
 begin
   logger.log_information(p_text => '(' || $$plsql_line || ') Start', p_scope => l_scope);
   -- write a debug message
   logger.log(p_text => '('|| $$plsql_line || ') ' || 'This is a debug message in '||l_scope
             ,p_scope => l_scope);
   -- write an information message
   logger.log_information(p_text => '(' || $$plsql_line || ') ' || 'This is an information message in '||l_scope
                         ,p_scope => l_scope);
   -- write a warning message
   logger.log_warning(p_text  => '(' || $$plsql_line || ') ' || '[' || sqlcode || ']' || sqlerrm ||
                                 'This is a warning message in '||l_scope
                     ,p_scope => l_scope
                     ,p_extra => dbms_utility.format_error_backtrace);
   -- write an error message
   logger.log_error(p_text  => '(' || $$plsql_line || ') ' || '[' || sqlcode || ']' || sqlerrm ||
                               'This is an error message in '||l_scope
                   ,p_scope => l_scope
                   ,p_extra => dbms_utility.format_error_backtrace);
   logger.log_information(p_text => '(' || $$plsql_line || ') End', p_scope => l_scope);
 end;
```

Same content

```
procedure this_needs_investigation is
   l_scope  logger_logs.scope%type := gc_scope_prefix || 'this_needs_investigation';
   l_params logger.tab_param;
 begin
   logger.log_information(p_text => '(' || $$plsql_line || ') Start', p_scope => l_scope);
   -- write a debug message
   logger.log(p_text => '('|| $$plsql_line || ') ' || 'This is a debug message in '||l_scope
             ,p_scope => l_scope);
   -- write an information message
   logger.log_information(p_text => '(' || $$plsql_line || ') ' || 'This is an information message in '||l_scope
                         ,p_scope => l_scope);
   -- write a warning message
   logger.log_warning(p_text  => '(' || $$plsql_line || ') ' || '[' || sqlcode || ']' || sqlerrm ||
                                 'This is a warning message in '||l_scope
                     ,p_scope => l_scope
                     ,p_extra => dbms_utility.format_error_backtrace);
   -- write an error message
   logger.log_error(p_text  => '(' || $$plsql_line || ') ' || '[' || sqlcode || ']' || sqlerrm ||
                               'This is an error message in '||l_scope
                   ,p_scope => l_scope
                   ,p_extra => dbms_utility.format_error_backtrace);
   logger.log_information(p_text => '(' || $$plsql_line || ') End', p_scope => l_scope);
 end;
```

Run the program in Debug mode [NoScope]

```
declare
  procedure set_log_level(level_in varchar2) is
  ...
  procedure set_scope_level(scope_in in varchar2
                           ,level_in in varchar2) is
  ...
begin
  set_log_level('DEBUG');
  logger_specific.run_all;
end;
/
```

```
select ll.logger_level lvl
      ,ll.text
      ,ll.scope
  from logger_logs ll
 where 1 = 1
   and trunc(ll.time_stamp) = trunc(systimestamp)
 order by ll.id asc
```

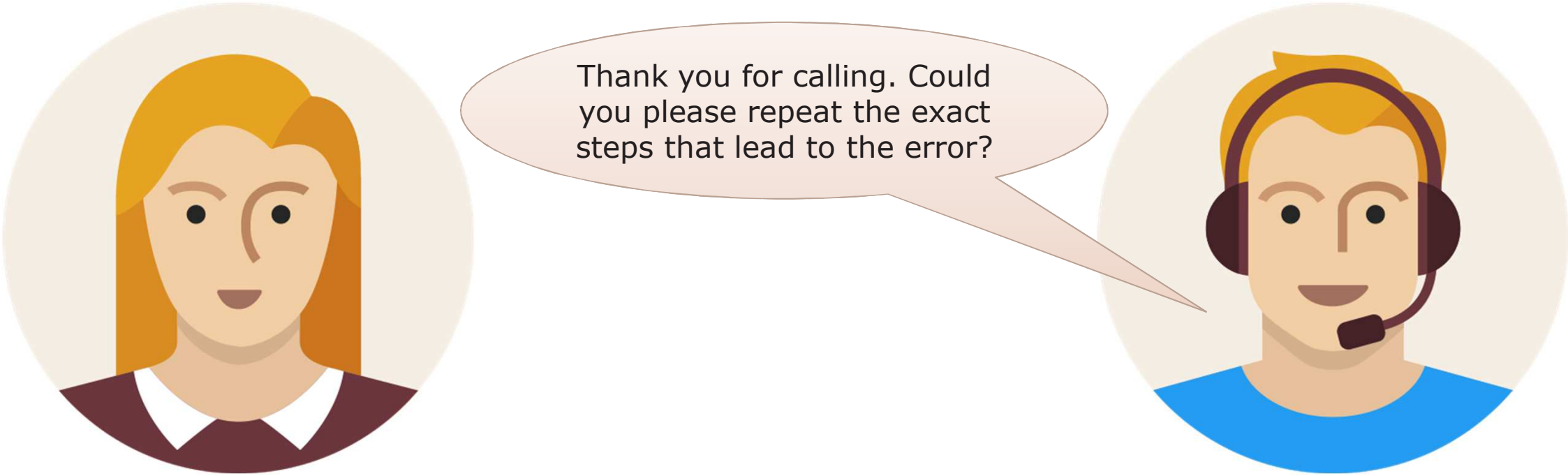
Run the program in Warning mode [Scope]

```
declare
  procedure set_log_level(level_in varchar2) is
  ...
  procedure set_scope_level(scope_in in varchar2
                           ,level_in in varchar2) is
  ...
begin
  set_log_level('WARNING');
  set_scope_level(user||'.LOGGER_SPECIFIC.JUST_SHOW_ERRORS’
                 ,'ERROR');
  set_scope_level(user||'.LOGGER_SPECIFIC.THIS_NEEDS_INVESTIGATION’
                 ,'DEBUG’);
  logger_specific.run_all;
end;
/
```

```
select ll.logger_level lvl
      ,ll.text
      ,ll.scope
  from logger_logs ll
 where 1 = 1
   and trunc(ll.time_stamp) = trunc(systimestamp)
 order by ll.id asc
```

```
LVL TEXT                                                                                           SCOPE
--- ---------------------------------------------------------------------------------------------- ---------------------------------------------
  4 (15) [0]ORA-0000: normal, successful completionThis is a warning message in demo.logger_specific.normal_behaviour        demo.logger_specific.normal_behaviour
  2 (19) [0]ORA-0000: normal, successful completionThis is an error message in demo.logger_specific.normal_behaviour        demo.logger_specific.normal_behaviour
  2 (39) [0]ORA-0000: normal, successful completionThis is an error message in demo.logger_specific.just_show_errors        demo.logger_specific.just_show_errors
  8 (49) Start                                                                                     demo.logger_specific.this_needs_investigation
 16 (51) This is a debug message in demo.logger_specific.this_needs_investigation                  demo.logger_specific.this_needs_investigation
  8 (53) This is an information message in demo.logger_specific.this_needs_investigation           demo.logger_specific.this_needs_investigation
  4 (55) [0]ORA-0000: normal, successful completionThis is a warning message in demo.logger_specific.this_needs_investigation demo.logger_specific.this_needs_investigation
  2 (59) [0]ORA-0000: normal, successful completionThis is an error message in demo.logger_specific.this_needs_investigation demo.logger_specific.this_needs_investigation
  8 (62) End                                                                                       demo.logger_specific.this_needs_investigation

9 rows selected
```

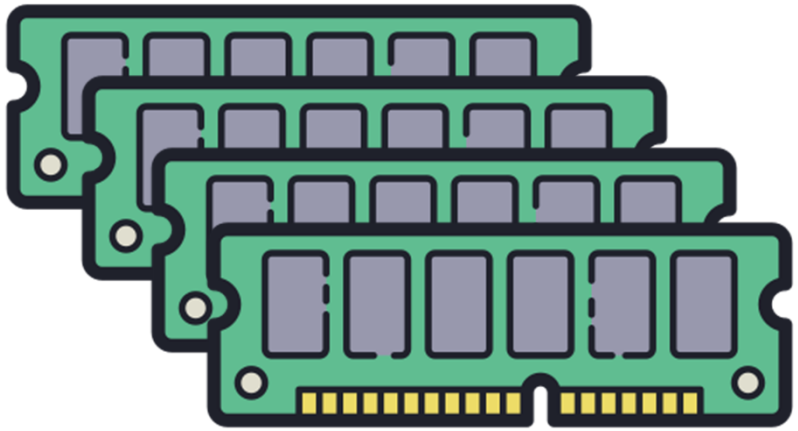

Hello helpdesk,
Something went terribly wrong. Can you please help me?

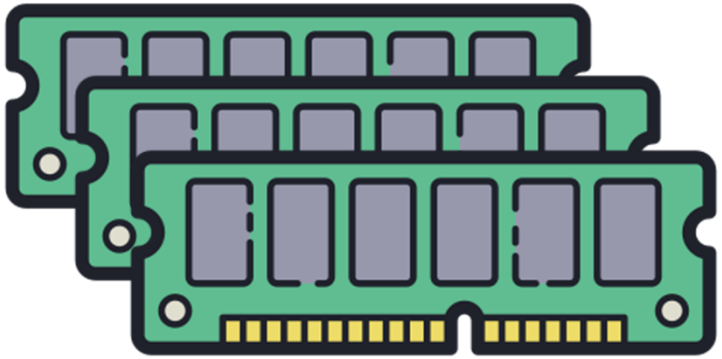

Thank you for calling. Could you please repeat the exact steps that lead to the error?


Do you expect me to memorize all the steps I take until my task completes successfully?

This way I can turn on the debugging program and get more information on what lead to this error.


There has got to be a better way to do this






Hello helpdesk,
Something went terribly wrong. Can you please help me?

Please hold while I check the logs for the source of the problem.


So you can see exactly what
steps I took in the program?

Only the last steps that lead
to this error. Nothing else.

Resources

- OraOpenSource - Your source for Oracle Open Source

  https://github.com/OraOpenSource

  https://github.com/OraOpenSource/Logger

- Log by Scope – Patch72 Fork of the Logger Framework

  https://github.com/patch72/Logger


Q&A

Oracle Cloud Infrastructure

New Free Tier

oracle.com/cloud/free

Always Free

Services you can use for unlimited time

+

30-Day Free Trial

Free credits you can use for more services