# 1. Basic

PL/SQL: Procedural Language/Structured Query Language là một ngôn ngữ lập trình hướng thủ tục dành cho Oracle SQL. Nó là mở rộng của Oracle SQL.

PL/SQL bao gồm các thành phần hướng thủ tục: vòng lặp, điều kiện.

PL/SQL cho phép khai báo hằng số, biến, thủ tục, hàm, kiểu dữ liệu và biến của các kiểu dữ liệu, trigger. Nó có thể xử lý các exception tại thời gian chạy.

PL/SQL còn có thể sử dụng mảng.

PL/SQL từ version 8 trở đi còn có thêm các tính năng hướng đối tượng.

Lệnh SELECT…INTO trả về nhiều dòng (>1) gây ra exception

Lệnh SELECT…INTO không trả về dòng nào (=0) cũng gây ra exception

Sử dụng **:=** để gán giá trị cho 1 biến

# 2. “=>”

=> nghĩa là gán giá trị cho parameter

Example:

DBMS\_AQ.ENQUEUE( queue\_name => REQUEST\_QUEUE\_NAME

, enqueue\_options => v\_enqueue\_options

, message\_properties => v\_msg\_props

, payload => p\_payload

, msgid => out\_message\_id);

dbms\_output.put\_line ('msg enqueued; corr\_id: ' || out\_correlation\_id || ' message\_id: ' || out\_message\_id );

Explain:

That is the keyword/value notation for passing parameters to a PL/SQL procedure or function.

The left side is the name of the parameter, the right is the value being passed.

It's useful when you don't want to keep to a specific ordering of parameters, or for self-documenting code.

Bên trái là name of parameter. Bên phải là value

Điều này hữu ích vì ta không cần phải nhớ order của các parameter.

# 3. dbms\_output.put\_line

Câu lệnh này in ra màn hình output thôi

# 4. Procedure và gọi hàm chạy trực tiếp:

## Example:

Below example can be used to run directly:

DECLARE – Có khai báo function là thành procedure, còn để không thế này là pl/sql

DECLARE

BEGIN

eh\_sal.sal2imx1('1503230003');

dbms\_output.put\_line('hehe');

END;

Ở đây eh\_sal là nằm trong packages, có đuôi là pck

sal2imx1 là PROCEDURE, được viết đại khái as below:

PROCEDURE sal2imx1(case\_id VARCHAR2) AS

ws\_request eh\_sal\_price\_query\_req;

ws\_response eh\_sal\_price\_query\_resp;

enqueue\_options DBMS\_AQ.enqueue\_options\_t;

dequeue\_options DBMS\_AQ.dequeue\_options\_t;

message\_properties DBMS\_AQ.message\_properties\_t;

message\_handle RAW(16);

request\_id NUMBER;

connectionTimeout NUMBER;

staticInterval NUMBER;

queueMaxWait NUMBER;

no\_messages exception;

pragma exception\_init (no\_messages, -25228);

BEGIN

*-- construct ws request*

debug.open(vcProgramName => c\_user||case\_id);

ws\_request := construct\_request(case\_id);

log(to\_CHAR(SYSDATE,'HH:MI:SS') || '-> CONSTRUCTED');

begin

select nvl(param\_val, 30)

into connectionTimeout

from v\_proc\_conf

where proc\_key = 'sal2imx1'

and param\_key = 'connectionTimeout'

and rownum < 2;

exception

when no\_data\_found then

connectionTimeout := 30;

end;

begin

select nvl(param\_val, 10)

into queueMaxWait

from v\_proc\_conf

where proc\_key = 'sal2imx1'

and param\_key = 'queueMaxWait';

exception

when no\_data\_found then

queueMaxWait := 10;

end;

begin

select nvl(param\_val, 2)

into staticInterval

from v\_proc\_conf

where proc\_key = 'sal2imx1'

and param\_key = 'staticInterval';

exception

when no\_data\_found then

staticInterval := 2;

end;

SELECT SEQ\_EH\_SAL\_REQ.nextval INTO request\_id FROM dual;

*-- push ws request in request queue. we use immediate option to make it vissible to listeners*

*-- independently from our transaction.*

enqueue\_options.visibility := DBMS\_AQ.IMMEDIATE;

message\_properties.correlation := to\_char(request\_id);

message\_properties.expiration := DBMS\_AQ.NEVER; *-- queueMaxWait;*

DBMS\_AQ.ENQUEUE(queue\_name => 'AQ\_EH\_SAL\_REQ',

Enqueue\_options => enqueue\_options,

Message\_properties => message\_properties,

Payload => ws\_request,

Msgid => message\_handle);

log(to\_CHAR(SYSDATE,'HH:MI:SS') || '-> RQ\_ENQUEUED');

*-- wait for ws response on the response queque*

BEGIN

dequeue\_options.wait := queueMaxWait + staticInterval; *-- the time is in sec*

dequeue\_options.deq\_condition := 'corrid = ' || to\_char(request\_id);

dequeue\_options.visibility := DBMS\_AQ.IMMEDIATE;

DBMS\_AQ.DEQUEUE(queue\_name => 'AQ\_EH\_SAL\_RESP',

dequeue\_options => dequeue\_options,

message\_properties => message\_properties,

payload => ws\_response,

msgid => message\_handle);

dbms\_output.put\_line('response waited');

EXCEPTION

WHEN no\_messages THEN

BEGIN

dequeue\_options.wait := 0; *-- the time is in sec*

dequeue\_options.deq\_condition := 'corrid = ' || to\_char(request\_id);

dequeue\_options.visibility := DBMS\_AQ.IMMEDIATE;

DBMS\_AQ.DEQUEUE(queue\_name => 'AQ\_EH\_SAL\_REQ',

dequeue\_options => dequeue\_options,

message\_properties => message\_properties,

payload => ws\_request,

msgid => message\_handle);

log(to\_CHAR(SYSDATE,'HH:MI:SS')||'-> No process waiting for the request.');

debug.close;

raise\_application\_error(-20113, eh\_sal.getTranslatedErrm(USERENV('LANG'), '44'));

EXCEPTION

WHEN no\_messages THEN

BEGIN

dequeue\_options.wait := connectionTimeout + staticInterval; *-- the time is in sec*

dequeue\_options.deq\_condition := 'corrid = ' || to\_char(request\_id);

dequeue\_options.visibility := DBMS\_AQ.IMMEDIATE;

DBMS\_AQ.DEQUEUE(queue\_name => 'AQ\_EH\_SAL\_RESP',

dequeue\_options => dequeue\_options,

message\_properties => message\_properties,

payload => ws\_response,

msgid => message\_handle);

EXCEPTION

WHEN no\_messages THEN

log(to\_CHAR(SYSDATE,'HH:MI:SS')||'-> SAL interface timeout.');

debug.close;

raise\_application\_error(-20113, eh\_sal.getTranslatedErrm(USERENV('LANG'), '45'));

END;

END;

END;

*-- process ws response*

IF ((ws\_response.err\_code IS NOT NULL) OR (ws\_response.err\_msg IS NOT NULL))

THEN

log(substr(to\_CHAR(SYSDATE,'HH:MI:SS')||'-> ' || ws\_response.err\_code || '; ' || ws\_response.err\_msg, 0, 300));

debug.close;

raise\_application\_error(-20100,'' || ws\_response.err\_code || '; ' || ws\_response.err\_msg);

ELSE

log(to\_CHAR(SYSDATE,'HH:MI:SS')||'-> RS\_DEQUEUED');

process\_tariff(ws\_response.response);

log(to\_CHAR(SYSDATE,'HH:MI:SS')||'-> END');

debug.close;

END IF;

*-- commentar zaradi ne6to se, DV ... Darina za pove4e info :-D*

*-- INSERT INTO t\_se\_interface*

*-- (refdoss, action, dt\_creation\_dt)*

*-- VALUES*

*-- (ws\_response.response.case\_id, 'AUTOMATIC TARIFF RESPONSE' , SYSDATE);*

END;