# Lec 1 PL/SQL

Composite data type

1. Record  carry variables with different data types

|  |  |  |
| --- | --- | --- |
| number | varchar() | date |

2- array  carry variables with the same data types

|  |  |  |
| --- | --- | --- |
| number | number | number |

Variable bb number  must be not written in ed , just write it in shell screen

To reference the bind variable in block or in select statement use :bb

Print variable  Used to print the bind variable print bb without : if you not in block or select statement . must first access set auto print on , Will print all blind variables that declared in the last block

Print  W/ill print all blind variables that declared in the last block

Set server output on  used to run DBMS\_OUTPUT.PUT\_LINE(); "used to print value in screen"

Scope of it "session"

NOT NULL variable must have initialize value . name varchar(20) NOT NULL :='ali'

DBMS\_OUTPUT.PUT\_LINE();  must be executed in ed " because it 's package carry function put.line() "

Procedure statement  if ,for ,variables assignment …………………..

Pl  Procedure statement && sql statement . anything between begin&& end

Procedure statement  can't access the DB so , you could't write any sql function that use db in it such as length(ename)

To acess length(empid)  you must write pure sql statement .

Select length(ename) from emp ;

To label the pl  write <<name>> AS declare <<name>> OR begin <<name>> or before declare.

Select ename into v\_name from emp;

# V\_name variable get data in ,cam be viewed by print

# Select must return one row only

# Number &&data types in select equal to number&&data type of valuables .

# To allow select to return more than one row use curser.

# comment -- or /\* ……\*/

# The variables name must be not as columns names .

# private memory area called context area ,contain info about the last sql statement that done .

# curser  pointer to context area.

# types of curser  implicit && explicit

Implicit  done by DBMA automatic when do any sql statement

Explicit done by user ,to the multi row elements return from select statement .

# cuser attributes  carry info about the last statement don

#row count number of row affected (o/p).

#found  the statement done

#notfound  عكس found

#sql% attri بيشاور على ال curser

Control structure:

Case - vi

Case  anci && can implement any condition

decode  oracle && implement only equality .

Declare

v\_rec departement %roe=wtype

begin

select \* into v\_rec from dep

v\_rec.coulmnname

v-rec is variable that carry many record coulmns with the same name &&data type of the table

&& you can call it using the record name.cloumn name

## Lec 2 PL/SQL (1-5 && 7,8,9)

Lesson 7

Cursors :

Cursor attribute not used in sql statement but used in procedure statement .

Controlling Explicit cursor:

Declare 🡪 open 🡪 fetch 🡪end check 🡪close

Declare 🡪 allocate memory for context area

Define structure of the query as "بتحجز مكان ليةبس فى الميمورى"

|  |  |  |
| --- | --- | --- |
| no | Name | Sal |
|  |  |  |
|  |  |  |

OPEN🡪 load data && allocate memory for data "do the query && put data into memory " allocate pointer to the first row.

|  |  |  |
| --- | --- | --- |
| no | Name | Sal |
| 1 | Shshssh | 10000 |
| 2 | Ddhd | 15151 |

Fetch 🡪 "take fetch the rows in the variables row by row

&& make advance to pointer to the next row ((row ++))"

END🡪 make check if the cursor empty or not to fetch another row or not "check on data "

Close 🡪 disable the curser && make release to active set "de-allocate memory assigned to data"

1. "Declaration" Declare cursor-name is

Any Select statement

انا لو عملت declare مش كل مرة هعملها دى بتتعمل مرة واحدة بس

1. Open&&fetch &&end &&close

Begin

Open cursor-name;

Loop //loop to get all rows of cursor

Fetch cursor-name into variable1,variable2 ,….

Note ,the variables must be equal to return of cursor in number&&data type .

&& then you can print this variable ;

Exit when cursor-name%notfound; //for check

Not use sql% 🡪 because it's explicit cursor that has data && name.

End loop;

Close cursor-name;

END;

Cursor for loops make open && fetch && close automatic &&exit condition && define the variables that I will fetch in

V\_rec cursor-name % row type

Declare curser

Begin

For I in cursor-name //the I with the same data type of curser.

Loop

Print I data

End loop ;

End;

Cursor Attributes

% found 🡪 return true if the row fetched found.

%row count🡪 get the number of rows affected "the I fetch it'

% is open 🡪 check that the cursor is open .

Cursor with parameter:

Send parameter to the cursor && the cursor will fetch data correspond to this parameter..

Curser name(parameter) is

Select \* from emp where deptno =parameter ;

Open name (10) ; open name(20);

For update clause

Curser read only can't be used for update, to do this

1. Declare curser name is

Select \* from ---- where --- for update ;

2) fetch the curser-data

3) UPDATE TABLE SET sal =10000

Where current of curser-name

Select statement 🡪 not return error if there's no o/p

### Lesson 8 Handling exception ;

Exception section between begin &&end ;

Exceptions types:

1- Predefined exc.

Implicitly raised by DBMS

Has oracle name and number 20 error.

2- NON-predefined exc.

Implicitly raised by DBMS

Has oracle number only . 20000 error

3- user defined exc

explicitly raised by user "or program"

Has no oracle name and number

Handling steps :

1- Predefined exc.

Handle the exception using name .

2- NON-predefined exc.

Name the exception "declare"

Associate name by the oracle number"اربطهم ببعض".

Handle the exception

3- user defined exc

Declare the exception by name .

Raise the exception .

Handle the exception

The non-predefined exc

Declare

Ex exception; //name & associate to number of exception .

PRAGMA EXCEPTION\_INIT(EX,-Number) ;

Begin

Exception section

End

-------------------

Sql error code 🡪number of the last exception

Sql error message🡪 name of the last exception

Both are access using procedure statement not the sql statement

The user-defined exception

Declare in the declaration section

Begin

RAISE exception-name ;

Exception section handle the exception

Sql%found 🡪 get true if the last statement done without any error .

Raise will pass me to the exception handle section .

If the inner exception handle not contain the error ,the error will move to exception section of the outer section to be handled , if not the exception will print on the output screen

If I need that the statement after the inner exception section to be done , you must be other statement to handle any exception , and move to the next statement to be run .

If I need sequence of statement to be run with other or not run if one of them failed , Make save point before begin these statement and in the exception section make roll back to the same save point

RAISE\_APPLICATION\_ERROR (-20121,"this when no data found") 🡪 will show the error of user defined exception as the oracle error form

Range of free numbers in error (20000,20999) , that can be used in to show error in specific way .

### Lesson 9

# Anynouns 🡪complied each time run it .

#Procedure && functions 🡪 compiled once then stored in DB

#Function must return a return one value or more

#Procedure must not return a return value

Procedure.

Mode default in

In 🡪 take the value in this parameter

Out 🡪put in the parameter value

In/out 🡪 take value && put new value

Create or replace procedure name

[AR1 mode type ,AR2 mode type]

IS | AS

Variables declarations

Begin

End;

كدا حتى الان وكانى عملت creation ليها وليس calling

To revoke the procedure "execute"

Begin

Procedure name;

End

OR

Sql> execute procedure name ;

OR

Sql> call procedure name ();

Function

Create or replace function name [parameters ]

Return data type 🡪 specify the returned data type

IS |AS

Function body ;

To revoke the function

Begin

Variable := Function name(attributes);

End

Show errors 🡪to show errors in the stored procedure

Data dictionary 🡪 info about data base tables && views && any thing in data It's table read only "metadata"

Tables begin with

1- user\_ "tables ,views, objects …." 🡪 any thing in my schema

2- all\_ ""🡺 any thing that I have access on it

3- dba\_ 🡪 any thing in schema of any person

4- v$ \_ 🡪 any thing related to the performance

5- Dictionary 🡪 contain all the name of the available data dictionary && comment on each one.

.

User\_scource 🡪table contain info about procedures