

## Overview of PLISOL

Introduction: Structured Query Language. (SQL) SQL is powerful language to DBA (DataBase Administrator). But it suffers with several disadvantages When used as a conventional programming language. SQL does not have procedural capabilities that means SOIL does not provide the basic programming constructs for condition checking, looping and biadring branching, that are essential for writing programs.

SQL (commands) are passed to the oracle engiene one at a time So, we cannot execute a group of sol commands. This adds additional traffic on the network, therefore decreasing the speed of data processing, specially in a multi-users environment. 3 Generally, oracle engine displays it's own error messages when processing SQL Statements and we cannot

Introduction to PL ISQL

customize error messages.

The PL (SQL programming language was developed by oracle corporation as a procedural extension of SQL commands. It supports the development of structures program and works as a bridge in between database and for front end tools.

世	PLISQL Black Structure:
	DECLARE
	L declaration-section>
	BEGIN
	L'executable commands>
-	EXEPTION
	END;
	Administration of the second o
	Fig: - A Generic Structure of PLISOLBlock.
#	The generic block of PLISOL program consist of the following PLISOL Block Structure sections:
_	The generic block of PLISOL program consists of the following sections:
C	DDECLARE:
-	The declaration section that starts with the
	keyword declare, declares all variables, constants,
	cursors, sub programs and other elements to be used in a program.
11.	
	ØBEGIN:
	The begin section is a compulsory section of all
	PLISOL programs. It consists of executable SOL and
	PLISOL Structures which describe the process that
ma C	

have to be applied to the tables. Actual data manipulation, data retrieval, looping, branching are spacified in this section. I are

(3) EXCEPTION:

This section deals with the handling of a during the execution of data manipulation states inside the begin section. It begins with the key exception and it is optional.

(4) END :

Every PLISQL block must be terminated (END; (semicolon)

It indicates the end of the program-

Variables: Variable in PLISQL begin with a charactery maximum 30 characters variables are the maximum. In PLICAL Locations used to had values. In PLISOL he assign some value to the variable using assign operator (:=) And, we can also fetch data Variables from data tables. Variables are declare Variable Name, Data type; for eg: balance FLOAT, pame VARCHAR (30); balance = 75405.50; SELECT S. name INO name > variable FROM Student WHERE REP. NO = 1001; Declaring a constant is similar to declare a variable except that the keyword constant mis be added to the variable name and a value assigned immediately. There after no further ass to the constants are possible. Syntax: Constant\_Name CONSTANT dataType = value balance CONSTANT FLOAT = 100.00;

Condition Control Structure: IF THEN-ELSEIF- ELSE-ENDIF

Syptax:

If (condition) THEN

action;

ELSE condition THEN\_

action;

END IF ;

for eg .

Assume that we have a table as:

EMP (EmpNo, EName, Post, Salary, Dept No) with

few records.

Now write a PLISQL block to take employee No as input, fetch the salary of that employee and increase the salary by 2000 if salary L= 20,000 Otherwise, increase the salary by 5,000.

Kogram:

DECLARE

E no VARCHAR (10):

Sal FLOAT;

new sal FLOAT;

BEGIN

eno:= 2 eno: OR eno:=: eno: /1 To

Take input from the user



# Types of Cursor:

Cursor are classified depending on the situations under which they are opened. There are two types of

1) Implicit Cursor.

2) Explicit & Cursor.

engiene itself.

This cursor is opened by Oracle engine for it's internal processing, everytime when an SOL Statement is processed. Since, the implicit cursor is opened and managed by oracle engine internally, the functions of reserving an area in memory, populating this area with appropriate data, processing the data in memory area when

the processing is complete is taken care by cracle

Attributes of Implicit Cursor

1) 1. ISOREN 3) 7. NOT FOUND

2) 7. FOUND 4) 7. ROWCOUNT.

Implicit Cursor have the following attributes:

1) · I Is OPEN: It returns true if cursor is Open. Otherwise false. It is written as SQL-1. ISOPEN.

- 2) 1- FOUND: It evaluate true if an INSERT, Del UPDATE Statement affect at least one record false. It is written as SQL-1-FOUND.
  - 3) 1. NOTFOUND: It is the legical opposite of 1. FOUND it evaluates true if insert, update and delete does not affect any record. Otherwise, false. It is written as SQLI NOTFOUND.
  - 4) 1. ROWCOUNT: It returns the number of vows affect by INSCRT, UPDATE, DELETE or SCIECT Statement I written as SQL 1. ROWCOUNT.
- # For eg: A PUSBL Block to change the department

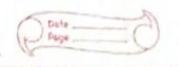
Emple, E. Name, Post, Salary, ... dept no)

eno Emp EmpNo 1- TYPE; dno Emp. DeptNo 1. TYPE:

eno:= : eno; UPDATE Emp SET Dept no = d.no WHERE Emp. No = em; If SOL I FOUND THEN dbms - autput put-line ("Employee Transferred Syccessfully



ELST dbms-output put-line ('Employee does not exist); END; g: A PLISOL Block to input department number and increase the salary by 5,000 for the employee of given department number and counting the number of records affected DECLARE dino Emp. Dept no: 1. TYPE; rows VARCHAR 2 (5); BEGIN dro := : dro; UPDATE Emp SET Salary = Salary + 5,000 WHERE Dept. no = dno; rows = TO-CHAR (SOL ! ROWCOUNT); IF SOL! ROWAUNT DO THEN doms-output put-line (raws it 'Rows affected'); ELSE about output put line ('No Employee in giver department END IF; END'



## Using PLISISL Subprograms.

Sub program A Sub program is a lagical group of SQL and PLISQL Statements that perform a specific task. These sub programs are combined to form larger programs. This is actually modular program design where a large and complex program is decomposed into multiple smaller and simple sub programs or modules. In PLISOL sub programs can be created as procedure and function. These sub programs can be called by passing a set of parameters. -> A PLISOL sub program (either procedure or function) is made up of three parts as: 1) Declarative Part. 2) Executable part. 3 Exception Handling (optional) 1) Delarative part The declarative part may contain the declaration of

these part is optional and does not start with the keyword declare.

variables, constants, cursor, exception and other sub

# Grecutable Part: This is the compulsory part and contains all SQL and PLISQL Statements necessary perform the desire tast. # Groption Handling Part: (optional) This is an optional part and it centains the code that hardles runtime errors of executable -> Write down the advantages of using Subprogram; 1 Improved Performance: The amount of information that: application must send over a network in Small Comparts with issuing individual SQL Statements or sending to text of an entire PLISOL block to oracle database, because the information is sent only once and there thereafter invoked when it is used. - The compiled form of a procedure is readily available the database. So, no compilation is required at execut \* Memory allocation: Because stored procedures take advantage of the shared memory capabilities of oracle database, it must load only a single copy of the procedure into memory for execution by multiple users sharing code among users results for agricultions in database memory requirement

# Passing Parameters: In PLISQL we can pars Paramox different ways as: 1) Positional method In positional notation, actual parame in function / procedure call must be in the same on function or practure definition. They must have ide datatypes. We cannot change the order of actual formal arguments. for Example: CREATE OR REPLACE FUNCTION Fun ( or Number Number, Z Number) END; function calling black BEGIN .. for stide today block, 8600 fun (b, c, a): In this case, the values of actual argument b, c and are copied to the formal arguments in the same our (2) Named Method: In this method we use an array operator while calling the function or procedure to pass the parameters. In this method, there is no fixed position for example:



CREATE OR REPLACE FUNCTION fun for Number, y Number, Z Number):

function calling block; BEGIN

fun (z ≥ a, x => b, y = 0; END;

3 Mixed Method:

In this approach few parameters are passed as positional and few using named method.

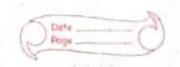
CREATE OF REPLACE FUNCTION for ( on Number, y Number, Z Number).

furction calling block;

fur (a, z ⇒ b, y ⇒ c); 11 legal fur (a > b, y ⇒ a, c); 11 illegal

END;

In mixed method. Pavameters on left of function call must be positional. So, we cannot write initial parameters as named followed by positional paramet



# Function

A PUSQL function is same as precedure except that it returns a value to the calling block. It is mainly used to accept arguments, to perform certain calculations and return a single value at a time.

A function is created as:

CREATE OR REPLACE FUNCTION function-name Cargument (IN] Datatype .....)
RETURN Datatype

& Is, AS3

Variable declaration;

BEGIN

PLISOL Subprogram body;

EXCEPTION

PLISOL Exception Handling block;

END;

where.

RETURN specifies the type of data returned by the function.



## # Private and Public Objects in a Pactage

The subprograms cursors, variables and constants declared inside the package body are private. Such private body are private. Such private objects are only accessible within the same package and cannot be accessed in PLISBL code outside the package.

Any subprograms, cursors, variables and constants declared within package specification are the public objects and become visible inside the same package as well as outside the package to access them outside the packages, we use

Package name object Name:

## # Overloading procedure & function

In oracle package we can declare and define more than one functions or procedure with the Same name but with different parameters or thoir datatypes. This concept is known as overloading functions or procedures. This concept is known as overloading overloading approach simplifies the PUSBL blocks by providing similar name to a class of functions or procedure that can perform similar types of task but on different types of arguments.



# Overview of Product Specific Packages

Oracle and various oracle tools are supplied

with product specific packages that defines API's

and we can call from PHSOL JAVA, and other

programming environment. Few commonly used packages are:

@ DBMS - ALERT:

This package allows database triggers to call an application or errors when specific database value changed. The alerts are transaction based and they operate independently. This package includes various methods to deal with different alert messages that may arise while performing or executing bransactions.

@ DBMS - OUTPUT: -

This package allows us to display output from PLISOL block and subprogram, which makes it easier to debug and test the PLISOL blocks. The method put. line() is used with this package for displaying some message

DBMS-PIPE:-

A pipe is an area of memory used by one process to pass information to other processes. This package allows different sessions to communicate over different pipes. This package is useful in various operating systems to interact with multiple high level language programs.



3 UTL FILE: read and write greating systems text file provides different file related operations like or PUTCH, GETCH, CLOSE (), etc to interact files for reading and writing operations.

SUR-HTTP:

This package allows PLISQL programs make hypatext transfer protocol call outs: it can retrieve data from the use pages Oracle web server as well as can provide to them. It exchanges data In the form

@ UTL-SMTP :

This package allows PLISQL programs to

D HTF and HTP:

Hypertext procedure (HIP) and Hyp Function (HJE) packages generate HTML tags to

Comesponding HTE functions HTML tags, there Consesponding HIE functions with identical prices