**TALLER 3**

Cristian Rios

David Acosta

Mayo 2018.

Universidad Ean.

Facultad de ingeniería.

Aplicativo de base de datos PL/SQL

**Abstract**

En el documento se describen algunos elementos y conceptos básicos del lenguaje PL/SQL para el posterior desarrollo de programas usando todos los conceptos que se describan

**Tabla de Contenidos**

[Funciones SQL Funciones realizadas sobre la base de datos 1](#_Toc511499150)

[Traducir 1](#_Toc511499151)

[Adición de algunas columnas 2](#_Toc511499152)

[**Borrado de algunas columnas**. 2](#_Toc511499153)

[**Poner opción default a algunas columnas**. 2](#_Toc511499154)

[**Modificar el tipo de datos y tamaño y/o Precisión/Escala de algunas columnas**. 3](#_Toc511499155)

[**Nombre del schema en Apex** 3](#_Toc511499156)

# Lenguaje PL/SQL

## Tipos de datos

Cada constante y variable tiene un tipo de dato en el que se especifica el formato de almacenamiento, restricciones y rango de valores válidos. PL/SQL proporciona una variedad predefinida de tipos de datos. A continuación, se muestran los tipos de datos más comunes:

* NUMBER (numérico): Almacena números enteros o de punto flotante, virtualmente de cualquier longitud, aunque puede ser especificada la precisión (número de dígitos) y la escala, que es la que determina el número de decimales.
* CHAR (carácter): Almacena datos de tipo carácter con un tamaño máximo de 32.767 bytes y cuyo valor de longitud por defecto es 1.
* VARCHAR (carácter de longitud variable): Almacena datos de tipo carácter empleando sólo la cantidad necesaria aún cuando la longitud máxima sea mayor.
* BOOLEAN (lógico): Se emplea para almacenar valores TRUE o FALSE.
* DATE (fecha): Almacena datos de tipo fecha. Las fechas se almacenan internamente como datos numéricos, por lo que es posible realizar operaciones aritméticas con ellas.
* Atributos de tipo. Un atributo de tipo PL/SQL es un modificador que puede ser usado para obtener información de un objeto de la base de datos. El atributo %TYPE permite conocer el tipo de una variable, constante o campo de la base de datos. El atributo %ROWTYPE permite obtener los tipos de todos los campos de una tabla de la base de datos, de una vista o de un cursor.

## Variables

Para programar en PL/SQL se pueden usar las variables de tipo PL/SQL (escalares o registros)

### Variables Escalares

Las variables escalares se usan para representar objetos de datos individuales de tamaño fijo como, por ejemplo, los enteros.

### Atributo Type

El atributo %TYPE sirve para declarar una variable a partir de otras declaradas previamente, o a partir del tipo de una columna de la base de datos. Es decir, copiar el tipo de otra variable.

El siguiente ejemplo asigna a la variable1 el mismo tipo que el campo de la base de datos:

variable1 tabla.campo%TYPE;

### Atributo ROWTYPE

se utiliza para declarar un registro con los mismos tipos que encuentre en la tabla, vista o cursor de la base de datos especificada.

## Constantes

Las constantes son como las variables, pero no puede modificarse su valor. Se declaran de la siguiente manera

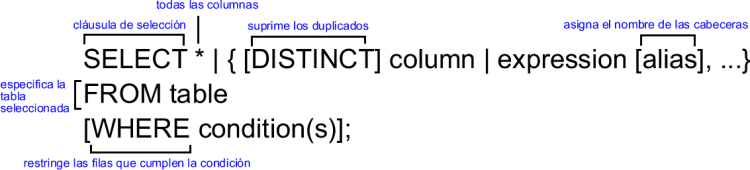
nombre\_constante CONSTANT tipo\_de\_dato := valor

## Sentencia Select

La sentencia SELECT se utiliza para recuperar información de la base de datos, y puede proyectar las columnas seleccionadas, es decir, realizar un filtro sobre la tabla o tablas originales y recuperar solamente datos de las columnas filtradas.

También nos permite filtrar los registros según varios criterios, o realizar uniones, que recopilan datos de diferentes tablas a través de una relación entre ellas.

El ejemplo de una estructura sencilla de una sentencia select en pl/sql es de la siguiente manera:

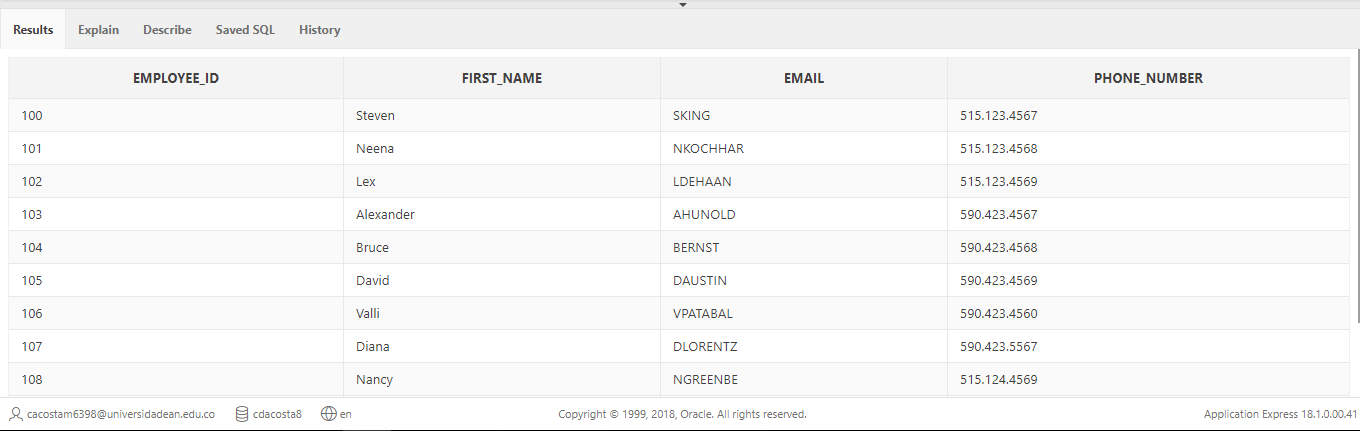


Ejemplo de una sentencia select

select \* from ibo.employees

### Seleccionar las columnas de una tabla

select EMPLOYEE\_ID,FIRST\_NAME,EMAIL,PHONE\_NUMBER from ibo.employees



Para seleccionar todas las columnas de una tabla se utiliza el \* tras el SELECT. Otra forma de hacerlo, sería ennumerar todas las columnas, separadas por comas, de la tabla como se muestra en el ejemplo anterior.

## Cursores

Un cursor está formado por un conjunto de registros devueltos por una instrucción SQL del tipo SELECT.

Desde un punto de visto interno a la base de datos Oracle, los cursores son segmentos de memoria utilizados para realizar operaciones con los registros devueltos tras ejecutar una sentencia SELECT.

Podemos distinguir dos tipos de cursores:

### Cursores implícitos

Este tipo de cursores se utiliza o se generan cuando ejecutamos una sentencia SELECT o cualquier sentencia de manipulación de datos.

### Cursores explícitos

Son los cursores que son declarados y controlados por el programador. Se utilizan cuando la consulta devuelve un conjunto de registros. Ocasionalmente también se utilizan en consultas que devuelven un único registro por razones de eficiencia. Son más rápidos.

Ejemplo estructura de un cursor explicito:

declare

cursor c\_paises (p\_continente IN VARCHAR2) is

SELECT CO\_PAIS, DESCRIPCION

FROM PAISES

WHERE CONTINENTE = p\_continente;

begin

/\* Sentencias del bloque ...\*/

end;

**Atributos de un cursor**

Para conocer detalles de la situación del cursor tenemos 4 atributos:

* %FOUND: devuelve verdadero di el ultimo FETCH ha recuperado algún valor; en caso contrario devuelve falso; si el cursor no esta abierto nos devuelve error.
* %NOTFOUND: hace justamente lo contrario al anterior.
* %ROWCOUNT: nos devuelve el número de filas recuperadas hasta el momento.
* %ISOPEN: devuelve verdadero si el cursor esta abierto.

### Partes de un cursor explicito

1. Declaración del cursor: lo tenemos que declarar en la zona de declaraciones, con el siguiente formato:

CURSOR <nombrecursor> IS <sentencia SELECT>;

1. Apertura del cursor: Deberá colocarse en la zona de instrucciones, con el siguiente formato:

OPEN <nombrecursor>;

1. Recogida de información: Para recuperar la información anteriormente guardada en las estructuras de memoria interna tenemos que usar el siguiente formato:

FETCH <nombrecursor> INTO {<variable> | <listavariables>};

Si tenemos una única variable que recoge los datos de todas las columnas, el formato de la variable seria el siguiente:

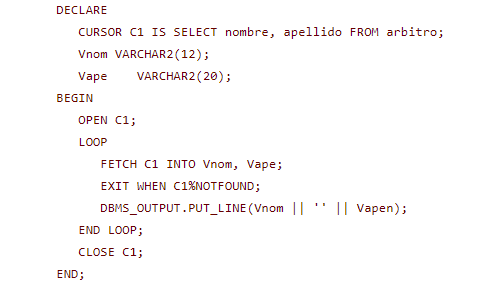
<variable> <nombrecursor>%ROWTYPE;

Si tenemos una lista de variables, cada una recogerá la columna correspondiente de la cláusula select, por lo que serán del mismo tipo que las columnas.

1. Cierre del cursor:

CLOSE <nombrecursor>;

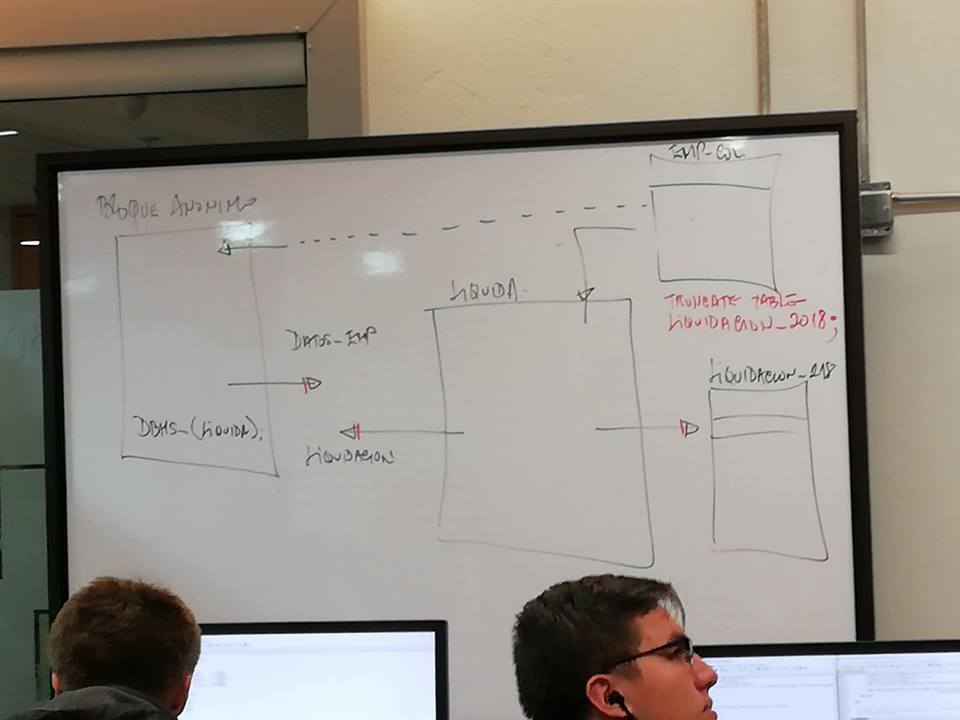
Ejemplo de dos cursores en funcionamiento:





# Problema propuesto Usando cursores y procedimientos almacenados

Realizar un programa que liquide todos los usuarios de una tabla colocando la logica que liquida a un usuario en un procedimiento almacenado. Este procedimiento sera ejecutado dentro de un cursor el cual le pasara registro a registro los datos de casa usuario para irlos liquidando y mostrando en pantalla .



1. Crear el Procedimiento que me permita capturar y realizar la liquidación de cada cliente :

CREATE OR REPLACE PROCEDURE LIQUIDACION\_USUARIO

(

U\_NOMBRE in VARCHAR2,

U\_APELLIDO in VARCHAR2,

U\_FECHA\_INGRESO in date,

U\_SALARY in FLOAT

)

AS

PRIMA\_DE\_SERVICIOS FLOAT := 0;

Cesantias FLOAT := 0;

Intereses\_sobre\_cesantias FLOAT := 0;

Vacaciones FLOAT := 0;

fecha\_inicioAnio date := to\_date('01/01/'|| to\_char(EXTRACT(YEAR FROM SysDate )) , 'dd-mm-yyyy') ;

BEGIN

PRIMA\_DE\_SERVICIOS := U\_SALARY \* (trunc(sysdate)-trunc(fecha\_inicioAnio)) /360 ;

Cesantias := (U\_SALARY \* (trunc(sysdate)-trunc(U\_FECHA\_INGRESO)) / 360);

Intereses\_sobre\_cesantias := Cesantias \* (trunc(sysdate)-trunc(U\_FECHA\_INGRESO)) \* 0.12/360 ;

Vacaciones := U\_SALARY \* (trunc(sysdate)-trunc(U\_FECHA\_INGRESO)) / 720 ;

DBMS\_OUTPUT.PUT\_LINE( U\_NOMBRE || ' ' || U\_APELLIDO || ' ........ Fecha Ingreso : ' || U\_FECHA\_INGRESO || ' Salario : ' || U\_SALARY );

DBMS\_OUTPUT.PUT\_LINE('Prima de servicios = ' || PRIMA\_DE\_SERVICIOS);

DBMS\_OUTPUT.PUT\_LINE('Cesantías = ' || Cesantias);

DBMS\_OUTPUT.PUT\_LINE('Intereses sobre cesantías = ' || Intereses\_sobre\_cesantias);

DBMS\_OUTPUT.PUT\_LINE('Vacaciones = ' || Vacaciones);

DBMS\_OUTPUT.PUT\_LINE('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*');

END;

1. Crear mi bloque anónimo que ejecute mi procedimiento almacenado pasándole la información de cada cliente

DECLARE

CURSOR C\_DEP IS SELECT \* FROM IBO.EMPLOYEES;

BEGIN

FOR VC\_EMP IN C\_DEP

LOOP

begin

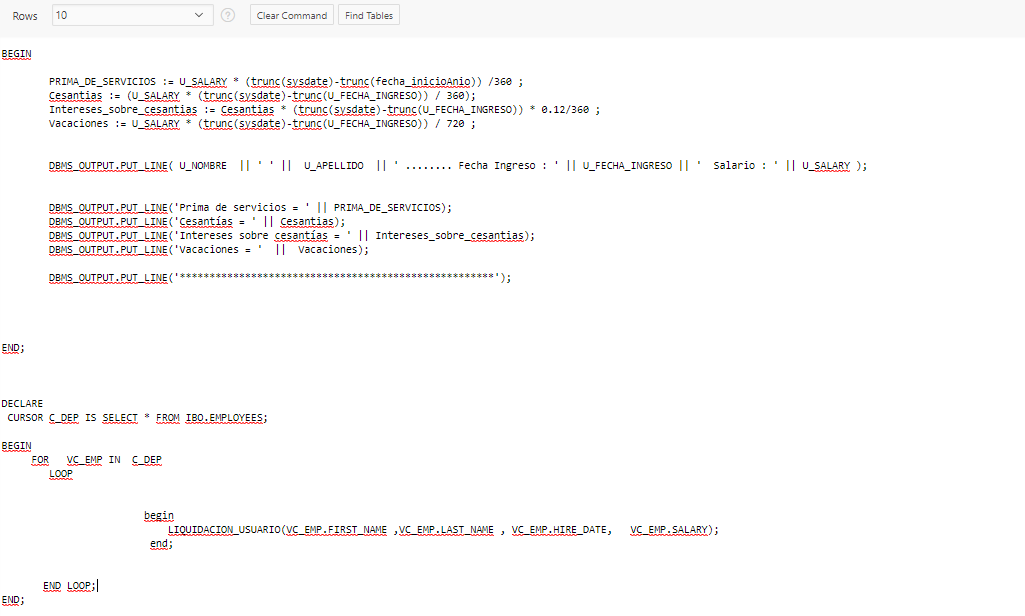
LIQUIDACION\_USUARIO(VC\_EMP.FIRST\_NAME ,VC\_EMP.LAST\_NAME , VC\_EMP.HIRE\_DATE, VC\_EMP.SALARY);

end;

END LOOP;

END;

1. Evidencia ejecución del código



1. Resultado

