

ÍNDICE

PRÓLOGO	XI
CAPÍTULO 1. FUNDAMENTOS DEL LENGUAJE PL/SQL	1
Introducción	1
Unidades léxicas.....	7
Tipos de datos	11
Declaración de variables	15
CAPÍTULO 2. ESTRUCTURAS DE CONTROL	21
Introducción	21
If-then-else-end if	22
Case	23
Loop...end loop	24
While...loop...end loop	25
For...loop...end loop	25
Goto	26
Null.....	27
CAPÍTULO 3. CONTROL DE TRANSACCIONES	29
Introducción	29

Commit.....	29
Rollback.....	33
Savepoint	37
Rollback to.....	37
CAPÍTULO 4. CREACIÓN DE TIPOS	41
Introducción.....	41
Creación de un tipo record (registro).....	41
Creación de un tipo table (pila de elementos).....	44
Varrays	49
CAPÍTULO 5. SQL vs PL/SQL	55
Introducción.....	55
Órdenes sql.....	55
Utilización de sql en pl/sql	57
Sql dinámico.....	61
CAPÍTULO 6. CURSORES	65
¿Qué es un cursor?	65
Cursores explícitos	66
Cursores implícitos.....	70
Cursores sql dinámico	71
CAPÍTULO 7. SUBPROGRAMAS	77
Introducción.....	77
Procedimientos	78
Funciones	86
New 12c Restringiendo permisos de uso a subprogramas.....	93
CAPÍTULO 8. PAQUETES	97
Introducción.....	97
Especificación o cabecera del paquete	98
Cuerpo del paquete.....	99
Referenciando a los paquetes	100
Inicialización de un paquete	100
Sobrecarga de paquetes.....	101
Dependencias.....	103
CAPÍTULO 9. PAQUETES PREDETERMINADOS	109
Introducción.....	109
Lista de paquetes predeterminados.....	109
Dbms_db_version	122

Dbms_file_transfer	123
Dbms_output	128
Dbms_random	133
New 12c Dbms_utility	138
Utl_file	159
Utl_mail.....	183
Utl_http.....	190
CAPÍTULO 10. DISPARADORES O TRIGGERS	243
Introducción.....	243
Utilidad de los triggers	244
Integridad referencial.....	244
Espacio de nombres del disparador	248
Momento del disparo	248
Suceso del disparo	249
Nivel de disparo	249
Condición de disparo	249
Sentencias de borrado y alteración de triggers.....	250
Uso de los predicados :old y :new.....	251
Uso de los predicados booleanos.....	251
Tablas mutantes.....	252
CAPÍTULO 11. TRATAMIENTO DE ERRORES	263
Introducción.....	263
Declaración de excepciones.....	264
Excepciones definidas por el usuario	264
Provocar excepciones	266
Sintaxis de la sección exception	267
Uso de sqlcode y sqlerrm.....	268
Utilización de raise_application_error	269
Utilización de exception_init.....	269
Propagación de excepciones.....	270
CAPÍTULO 12. CURSORES AVANZADOS	277
Bucles de extracción	277
Bucles simples (loop ... end loop).....	277
Bucles while	278
Bucles for	279
Cursores select for update	280
CAPÍTULO 13. OBJETOS	285
Introducción.....	285

Bases de la programación orientada a objetos	285
Objetos e instancias de los objetos	287
Bases de datos objeto-relacionales	288
Definición de los tipos de objetos	288
Llamada a un método	292
Borrar un objeto.....	292
Modificar un objeto	292
Creación de tablas de objetos	293
Inserción de valores en una tabla de objetos.....	293
CAPÍTULO 14. ENTORNOS DE EJECUCIÓN PL/SQL	295
Introducción	295
Sql*plus / isql*plus / sql*worksheet	295
Ejecución de código sql	296
Ejecución de código pl/sql.....	296
Definición de variables globales	297
Uso de variables globales	297
Cómo se puede llamar a un procedimiento almacenado.....	298
Cómo se puede llamar a una función almacenada	298
Envío de resultados a un archivo	299
Ejecución de scripts (archivos) de comandos.....	300
Mostrar errores de compilación.....	300
Herramientas de diseño	301
Oracle SQL developer.....	304
CAPÍTULO 15. CERTIFICACIONES DE ORACLE	309
Introducción	309
Certificaciones de Oracle disponibles	309
Preguntas tipo examen de certificación SQL.....	311
ANEXO I. RESOLUCIÓN DE SUPUESTOS PRÁCTICOS	335
Supuesto práctico 0.....	335
Supuesto práctico 1.....	337
Supuesto práctico 2.....	338
Supuesto práctico 3.....	340
Supuesto práctico 4.....	343
Supuesto práctico 5.....	348
Supuesto práctico 6.....	352
Supuesto práctico 7.....	355
Supuesto práctico 8.....	357
Supuesto práctico 9.....	358

Supuesto práctico 10	360
Supuesto práctico 11	364
Supuesto práctico 12	366
Supuesto práctico 13	368
Supuesto práctico 14	371
Supuesto práctico 15	373
ANEXO II. RESOLUCIÓN CUESTIONES DE CERTIFICACIÓN	377
Cuestión 1	377
Cuestión 2	378
Cuestión 3	378
Cuestión 4	379
Cuestión 5	379
Cuestión 6	380
Cuestión 7	380
Cuestión 8	381
Cuestión 9	382
Cuestión 10	383
Cuestión 11	383
Cuestión 12	384
Cuestión 13	384
Cuestión 14	384
Cuestión 15	385
Cuestión 16	386
Cuestión 17	386
Cuestión 18	387
Cuestión 19	387
Cuestión 20	387
Cuestión 21	388
Cuestión 22	388
Cuestión 23	389
Cuestión 24	389
Cuestión 25	390
Cuestión 26	390
Cuestión 27	391
Cuestión 28	391
Cuestión 29	391
Cuestión 30	392
Cuestión 31	392
Cuestión 32	393
Cuestión 33	394

Cuestión 34	394
Cuestión 35	394
Cuestión 36	395
Cuestión 37	396
Cuestión 38	396
Cuestión 39	396
Cuestión 40	397
Cuestión 41	397
Cuestión 42	397
Cuestión 43	398
Cuestión 44	398
Cuestión 45	398
Cuestión 46	399
Cuestión 47	399
Cuestión 48	399
Cuestión 49	400
Cuestión 50	400
Cuestión 51	400
Cuestión 52	400
Cuestión 53	401
Cuestión 54	401
Cuestión 55	401
ANEXO III. FICHERO SCRIPT_BDHOSPITAL	403
Introducción	403
Diagrama entidad/relación del esquema del hospital	404
Modelo relacional del esquema del hospital	405
Contenido del fichero con comentarios	405
ANEXO IV. REFERENCIAS Y MATERIAL ANEXO EN INTERNET	429
Referencias utilizadas para el curso	429
Enlaces a Oracle	429
ANEXO V. GUIA DE INSTALACIÓN DE ORACLE 11G XE	431
Introducción	431
Requerimientos mínimos	431
Tutorial de instalación	432
ÍNDICE ALFABÉTICO	439

FUNDAMENTOS DEL LENGUAJE PL/SQL

1

INTRODUCCIÓN

PL/SQL es un sofisticado lenguaje de programación que se utiliza para acceder a bases de datos Oracle desde distintos entornos. PL/SQL está integrado con el servidor de base de datos, de modo que el código puede ser procesado de forma rápida y eficiente. También se encuentra disponible en varias de las herramientas de cliente que posee Oracle, entre ellas SQL*PLUS, Developer Suite, JDeveloper, etc.

Si nos preguntamos por qué utilizar PL/SQL, la conclusión la encontramos en el propio SQL. Tenemos que recordar que Oracle es una base de datos relacional, que utiliza como lenguaje de datos el propio SQL. SQL es un lenguaje flexible y eficiente, con características muy potentes para la manipulación y examen de los datos relacionales, pero que presenta deficiencias a la hora de realizar programaciones procedimentales.

SQL es un lenguaje de cuarta generación (4GL), que como el resto de lenguajes de esta generación, presenta como característica el hecho de que describen lo que debe hacerse, pero no la manera de llevarlo a cabo. Por ejemplo si analizamos la siguiente instrucción:

```
DELETE FROM estudiantes WHERE nombre like 'Pep%'
```

Esta instrucción determina que queremos borrar de la tabla estudiantes todos aquellos cuyo nombre comience por "Pep", pero no dice cómo va a realizar el gestor de base de datos el proceso para conseguir eliminar dichos registros. Parece presumible que recorrerá los datos de dicha tabla en un cierto orden para determinar qué elementos debe borrar y luego los eliminará; no obstante, es algo que no nos interesa para la instrucción.

En contraposición a los lenguajes 4GL nos encontramos con los lenguajes de tercera generación (3GL), como C y Visual Basic. Son lenguajes más procedimentales donde se implementan algoritmos para resolver unos problemas. Estas estructuras procedimentales y de ejecución paso a paso no se pueden implementar en SQL, así que Oracle necesitaba de un lenguaje que pudiese resolver este tipo de problemas y que estuviera más enfocado no solo al manejo de datos sino a la resolución de problemáticas de todo tipo, así que creó el lenguaje PL/SQL (Lenguaje Procedimental / SQL). Este lenguaje no es solo un lenguaje de tipo 3GL, sino que permite utilizar la flexibilidad de SQL como lenguaje de 4GL.

Esta característica que lo define, es posible dado que es un lenguaje particular del sistema gestor de bases de datos Oracle y no un lenguaje estándar.

Por tanto el lenguaje PL/SQL potencia el lenguaje SQL agregando estructuras y objetos del siguiente tipo:

- El bloque
- Manejo de errores y excepciones
- Creación de procedimientos y funciones
- Definición de variables y tipos
- Estructuras de bucle
- Cursores
- Objetos

El bloque

Es la unidad básica de todo programa en PL/SQL. Todo programa al menos debe poseer un bloque.

Todo bloque consta de una sección declarativa optativa, una sección de ejecución obligatoria y una sección de control de errores optativa.

La sintaxis es la siguiente:

```
[DECLARE]
    <sección declarativa>
BEGIN
    <sección de ejecución>
[EXCEPTION]
    <sección de control de errores>
END;
```

SECCIÓN DECLARATIVA

En esta sección se definen las variables, constantes y cursores que se van a utilizar dentro de la sección de ejecución.

SECCIÓN DE EJECUCIÓN

La sección de ejecución presenta las siguientes particularidades:

- Toda instrucción `SELECT` (no dinámica) que aparezca en esta sección, deberá llevar incorporado el parámetro `INTO`, como se muestra en el siguiente ejemplo:

```
SELECT columnas INTO variables FROM tablas WHERE criterios.
```

- En esta sección solo se admiten instrucciones de SQL del tipo DML (select, insert, update y delete) o instrucciones SQL dinámicas, el resto no están permitidas implementarlas directamente (por ejemplo: alter, create, drop, etc.), salvo que se indique dentro de la instrucción `EXECUTE IMMEDIATE`.

SECCIÓN DE CONTROL DE ERRORES

En esta sección se definen los controles programados para detectar los errores de ejecución del bloque y la solución adoptada para los mismos.

Tipos de bloque

Se diferencian 3 tipos de bloques:

- Los *bloques anónimos* se construyen de manera dinámica y se ejecutan una sola vez.
- Los *bloques nominados* se construyen identificándolos con un nombre. Al igual que los anteriores, se construyen de forma dinámica y se ejecutan una sola vez.
- Los *subprogramas* son bloques nominados que se almacenan en la base de datos. Nos podemos encontrar con procedimientos, paquetes y funciones de este tipo. Siempre se ejecutan bajo demanda.
- Los *disparadores* son también bloques nominados que se almacenan en la base de datos, pero que no se pueden ejecutar bajo petición de un programa. Se ejecutan cuando tiene efecto el suceso para el que se han programado contra una cierta tabla del sistema.

EJEMPLO DE BLOQUE ANÓNIMO

```
DECLARE
    Var1 NUMBER;
BEGIN
    Var1 := 1;
END;
```

EJEMPLO DE BLOQUE NOMINADO

```
<<Nombre_Bloque>>
DECLARE
    Var1 NUMBER;
BEGIN
    Var1 := 1;
END;
```

EJEMPLO DE BLOQUE SUBPROGRAMA

```
CREATE OR REPLACE PROCEDURE MI_PROGRAMA IS
    Var1 NUMBER;
BEGIN
    Var1 := 1;
END;
```

EJEMPLO DE DISPARADOR

```
CREATE OR REPLACE TRIGGER MI_DISPARADOR IS
  BEFORE INSERT OR UPDATE OF numero ON tabla_temporal
  FOR EACH ROW
BEGIN
  IF :new.numero < 0 THEN
    RAISE_APPLICATION_ERROR(-20100, '!!!!Error!!!!');
  END;
```

Manejo de errores y excepciones

El lenguaje PL/SQL, como muchos otros procedimentales, permite un control sobre los errores que se produzcan en la ejecución del código. Esta gestión de errores presenta como ventaja la claridad para su manipulación, dado que utiliza una sección independiente a la del código ejecutable.

Creación de procedimientos y funciones

El lenguaje PL/SQL permite la creación de procedimientos almacenados y de funciones que nos devuelvan un valor como resultado de su ejecución.

Definición de variables y tipos

El lenguaje PL/SQL también permite la definición de variables para utilizar en nuestros programas y crear tipos de usuarios a partir de otros predefinidos.

Estructuras de bucle

Para desarrollar nuestros programas y poder realizar operaciones de bifurcación, el lenguaje PL/SQL posee estructuras de control.

Cursores

Este tipo de estructura, que se define en la sección declarativa de un bloque, nos permite recuperar en memoria un conjunto de filas de una tabla que se recorren una a una para un tratamiento posterior.

Objetos

Oracle dada la tendencia actual de los lenguajes de programación que hay en el mercado, incorpora también la figura del objeto, de forma que PL/SQL se puede convertir también en un lenguaje orientado a objetos.

Salida por pantalla de los resultados de una ejecución

PL/SQL a través de las herramientas propias de Oracle: SQL*Plus y SQL*Plus Worksheet únicamente visualiza el resultado satisfactorio o no de la ejecución de las instrucciones. Para poder realizar una visualización en pantalla de la ejecución y resultados internos del código PL/SQL hay que utilizar la invocación de un paquete incluido en Oracle PL/SQL denominado DBMS_OUTPUT, el cual permite dirigir a pantalla resultados mediante el uso de la función PUT_LINE.

No obstante para que se haga completamente efectiva dicha salida a pantalla, antes hay que activar el comando de ejecución en pantalla del paquete DBMS_OUTPUT, siempre que se inicie una nueva sesión con la herramienta correspondiente. Este comando es el siguiente:

```
SET SERVEROUTPUT ON;
```

EJEMPLO DE UN BLOQUE CON SALIDA A PANTALLA DE LOS RESULTADOS

```
SET SERVEROUTPUT ON;
DECLARE
    Var1 VARCHAR2(50);
BEGIN
    Var1 := 'Antolin';
    DBMS_OUTPUT.PUT_LINE('El nombre del autor es: ' ||
                          Var1);
END;
/
```

UNIDADES LÉXICAS

Es el conjunto de caracteres y gramática a utilizar en la programación de PL/SQL. Contamos con los siguientes elementos:

- Identificadores
- Palabras reservadas
- Delimitadores
- Literales
- Comentarios

Identificadores

Dan nombre a los distintos objetos de nuestro programa. Por ejemplo, los identificadores de variable, cursores, tipos, etc.

Un identificador tiene que comenzar obligatoriamente con una letra seguida por una secuencia de caracteres, entre los que se pueden incluir:

- Letras
- Números
- El símbolo \$
- El carácter de subrayado _
- El símbolo #

La longitud máxima de un identificador es de 30 caracteres.

También hay que tener en cuenta que el lenguaje PL/SQL no es un lenguaje sensible en cuanto a los caracteres, de manera que no distingue mayúsculas o minúsculas, salvo que el nombre vaya encerrado entre comillas dobles ("").

Cada objeto tiene un espacio de nombres.

Los espacios de nombres de objeto aglutinan una serie de nombres de objeto, en algunos casos de forma independiente, y en otros común para distintos nombres de objetos diferentes.



Fig. 1-1 Namespaces (Esquemas de nombre) de objetos de un esquema.

Dentro del mismo espacio de nombres no se puede repetir un nombre de objeto aunque sea de distinto tipo.

Por ejemplo, una tabla y una vista no se pueden denominar con el mismo nombre, porque comparten el mismo esquema de nombres. En cambio una restricción y un trigger sí se pueden llamar igual, porque cada uno de estos objetos del esquema de base de datos poseen esquemas independientes.



Fig. 1-2 Namespaces de objetos que no pertenecen a un esquema.

Ejemplos de nombres de esquema válidos son los siguientes:

```
EMP  
"Emp"  
SCOTT.FECHAALTA  
"INCLUSO ESTO & VALE!"  
UN_NOMBRE_LARGO_Y_VALIDO
```

Palabras reservadas

Son todas aquellas que define Oracle como restringidas dentro del lenguaje PL/SQL y cuyo nombre no podrá llevar ningún otro objeto de la base de datos.