CIFP CARLOS III

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

Tarea Unidad Trabajo 6

Módulo Bases de Datos

1º Desarrollo Aplicaciones Web

Fecha límite de entrega:

Domingo 10 de Mayo de 2020, 23:55 horas. Los envíos realizados a partir de ese momento llevarán una penalización del 30%.

Enunciado.

En la plataforma se tiene de nuevo un archivo (*discos.sql*) con una base de datos para almacenar la información del mercado musical. Es, exactamente, la misma que la utilizada en la tarea de la UT4. Se vuelve a disponer de las mismas 4 tablas cuyo significado sigue siendo el siguiente:

a. Tabla **DISCOGRAFICAS:** Almacena información sobre las casas discográficas que publican los discos

b. Tabla **GRUPOS**: Contiene la información de los grupos musicales. Se supone que cada grupo está asociado en la actualidad a alguna DISCOGRAFICA.

c. Tabla **MUSICOS**: Contiene la información personal de los músicos. Se supone que cada músico forma parte de un GRUPO

d. Tabla **DISCOS**: Contiene la información de los discos publicados por los grupos, así como de los millones de copias que se han vendido. Se supone que cada disco está asociado a un GRUPO y a una DISCOGRAFICA.

CIFP CARLOS III
C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

Tabla DISCOGRAFICAS:

Campo	Tipo	Referencia a:
cod_discografica*	number (4) not null	
nombre_empresa	varchar2 (20) not null	
pais	varchar2 (20)	
capital_social	number (12) not null	
tipo	varchar2 (3) not null,	

Tabla GRUPOS:

Campo	Tipo	Referencia a:
cod_grupo*	number (4) not null	
nombre_grupo	varchar2 (40) not null	
pais	varchar2 (20)	
fecha_fundacion	date not null	
fecha_disolucion	date	
estilo	varchar2 (40) not null	
discografica	number (4) not null	discograficas

Tabla MUSICOS:

Campo	Tipo	Referencia a:
cod_musico*	number (4) not null	
nombre_musico	varchar2 (30) not null	
pais	varchar2 (20)	
fecha_nacimiento	date	
grupo	number (4)	grupos

Tabla DISCOS:

Campo	Tipo	Referencia a:
cod_disco*	number (4) not null	
nombre_disco	varchar2 (40) not null	
fecha_publicacion	date not null	
millones_vendidos	number (10)	
formato_inicial	varchar2 (3)	
grupo	number (4)	grupos
discografica	number (4)	discograficas

Los campos marcados con asterisco (*) son los campos clave de las tablas.

CIFP CARLOS III
C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

Lo que hay que hacer.

El objetivo de esta práctica es la construcción de un **paquete SQL** llamado ***Gestion_Discografica***.

Ese paquete debe tener los siguientes procedimientos:

1.- Un procedimiento llamado INSERTAR_MUSICO que permita insertar un nuevo músico en la tabla MUSICOS. Recibirá los siguientes parámetros:

- El código que se asignará al músico (*cod_musico*)
- Nombre del músico (*nombre_musico*)
- País del músico (*pais*)
- Fecha de nacimiento del músico (*fecha_nacimiento*)
- Código del grupo al que está vinculado (*grupo*)

Este procedimiento INSERTAR_MUSICO ha de comprobar 2 cosas:

- El código de músico (*cod_musico*) que se recibe no existe ya en la tabla de MUSICOS
- El código de grupo (*cod_grupo*) al que el músico pertenece existe en la tabla GRUPOS

Si ambas condiciones se cumplen, el procedimiento debe realizar la inserción del nuevo músico en la tabla MUSICOS.

Si la primera condición no se cumpliese, se debe visualizar por consola el mensaje "ESE CODIGO DE MUSICO YA EXISTE. PRUEBA CON OTRO".

Si la segunda condición no se cumpliese, se debe visualizar por consola el mensaje "ESE CODIGO DE GRUPO NO EXISTE".

CIFP CARLOS III
C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

2.- Un procedimiento llamado TRABAJOS_HECHOS que recibe el *cod_musico* de un músico y visualice por consola la siguiente información:

- Nombre del músico (*nombre_musico*)
- Nombre del grupo musical al que pertenece (*nombre_grupo*)
- El nombre (*nombre_disco*) de todos los discos publicados por ese grupo ordenados cronológicamente por *fecha_publicacion*.

Este procedimiento TRABAJOS_HECHOS ha de comprobar que el código de músico (*cod_musico*) que se recibe existe en la tabla MUSICOS. Si ese *cod_musico* no existiese, se debe visualizar por consola el mensaje "MUSICO DESCONOCIDO".

3.- Un procedimiento llamado N_DISCOS que recibe 2 argumentos:

- Una *fecha* en formato DD/MM/AA
- Un número n

El procedimiento debe visualizar por consola los nombres de los n discos (*nombre_disco*) publicados (*fecha_publicacion*) con anterioridad a la fecha proporcionada. Se deben visualizar ordenados cronológicamente.

Si sucediera que el número de discos que se deben visualizar fuese inferior a n (Porque hay menos de n discos con fecha de publicación anterior a la fecha proporcionada), entonces no debe desencadenarse ningún error: Simplemente se visualizan los discos con fecha anterior a fecha.

CIFP CARLOS III
C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

3.- Un disparador llamado AUDITAR_DISCOS que va a llevar un control de las modificaciones, borrados e inserciones en la tabla DISCOS. Para ello se dispone de la tabla auxiliar AUDITORÍA, en la cual habrá que insertar un registro por cada operación llevada a cabo sobre la tabla libros. El disparador **no** se introduce en el paquete ***Gestion_Discografica.***

TABLA AUDITORÍA		
NOMBRE COLUMNA	**TIPO**	**DESCRIPCIÓN**
AUD_ID	Number	Clave primaria de la Tabla auditoría
AUD_FECHA	Fecha	Fecha en la que se guarda el registro
AUD_DISCO	Number(4)	Código del disco afectado
AUD_OPERACION	Cadena	Operación llevada a cabo: Inserción, borrado o Modificación.

CIFP CARLOS III
C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

(Las ayudas que se muestran a continuación son para trabajar desde línea de comandos, es decir, desde SQL Command. Para hacerlo con el ápex o con SQLDeveloper es exactamente igual, omitiendo la / final).

AYUDA 1:

Para crear procedimientos desde la línea de comandos de SQL se sigue la sintaxis mostrada en el siguiente ejemplo:

```
CREATE OR REPLACE PROCEDURE HOY_ES AS
BEGIN
-- Para visualizar la fecha de hoy:
  DBMS_OUTPUT.PUT_LINE( 'Hoy es ' || TO_CHAR(SYSDATE, 'DL') || '. Un gran dia' );
END HOY_ES;
/
```

Obsérvese que terminamos con el símbolo /.

Además, para visualizar por consola utilizamos el paquete DBMS_OUTPUT.

Con esto hemos creado el procedimiento llamado **HOY_ES**. Para invocarlo desde la propia consola, haremos:

```
BEGIN
  HOY_ES (); -- Los paréntesis son opcionales, pues no enviamos argumentos.
END;
/
```

AYUDA 2:

Para crear procedimientos que reciban argumentos (Tal y como sucede en los ejercicios de esta tarea), se sigue la sintaxis mostrada en el siguiente ejemplo:

```
CREATE OR REPLACE PROCEDURE PRODUCTO (p1 NUMBER, p2 NUMBER) AS
BEGIN
-- El producto de los 2 argumentos es:
  DBMS_OUTPUT.PUT_LINE( 'El producto es ' || p1*p2);
END PRODUCTO
/
```

Obsérvese que terminamos, de nuevo, con el símbolo /.

Además, para visualizar por consola utilizamos, de nuevo, el paquete DBMS_OUTPUT.

Con esto hemos creado el procedimiento llamado **PRODUCTO**. Para invocarlo desde la propia consola, haremos:

```
BEGIN
 PRODUCTO (12, 15);
END;
/
```

AYUDA 3:

Para crear un paquete con procedimientos (Que es exactamente lo que pide la tarea) se sigue la sintaxis mostrada en el siguiente ejemplo. Tal y como queda explicado en el contenido de la Unidad 6, es necesario especificar por separado la declaración del paquete y la construcción del mismo (Cuerpo):

```
-- Declaramos el paquete con sus 2 procedimientos:
CREATE OR REPLACE PACKAGE Ejemplo AS  -- Declaración del paquete

  PROCEDURE HOY_ES;
  PROCEDURE PRODUCTO (p1 NUMBER, p2 NUMBER);

END Ejemplo;
/


-- Construimos el paquete con sus 2 procedimientos:
CREATE OR REPLACE PACKAGE BODY Ejemplo AS  -- Construcción del paquete


-- Primer procedimiento:
PROCEDURE HOY_ES AS
BEGIN
-- Para visualizar la fecha de hoy:
  DBMS_OUTPUT.PUT_LINE( 'Hoy es ' || TO_CHAR(SYSDATE, 'DL') || '. Un gran dia' );
END HOY_ES;

-- Segundo procedimiento:
```

C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

```
PROCEDURE PRODUCTO (p1 NUMBER, p2 NUMBER) AS
BEGIN
-- El producto de los 2 argumentos es:
  DBMS_OUTPUT.PUT_LINE( 'El producto es ' || p1*p2);
END PRODUCTO;

END Ejemplo;
/
```

Si ahora quisiéramos invocar desde la consola a cualquiera de los procedimientos del paquete **Ejemplo** podríamos hacer, entre otras muchas cosas:

```
DECLARE
  valor1 NUMBER;
  valor2 NUMBER;

BEGIN

  valor1:= 37;
  valor2:= 43;

  Ejemplo.PRODUCTO (valor1, valor2); -- Invocamos el procedimiento PRODUCTO
  Ejemplo.PRODUCTO (101, 9); -- Invocamos el procedimiento PRODUCTO
  Ejemplo.HOY_ES; -- Invocamos el procedimiento HOY_ES
END;
/
```

CIFP CARLOS III
C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

AYUDA 4:

También podemos crear (dentro y fuera de los paquetes) funciones que, a diferencia de los procedimientos, siempre devuelven un valor. Por ejemplo, si quisiéramos añadir una función que reciba 3 números como argumento y devolviese el mayor de todos, lo haríamos así:

```
CREATE OR REPLACE FUNCTION MAYOR_DE_3 (n1 NUMBER, n2 NUMBER, n3 NUMBER)

   RETURN NUMBER AS -- La función devuelve un NUMBER

   mayor NUMBER;  -- Variable local a la función

   BEGIN

   if (n1 >= n2)
     then mayor := n1;
    else
         mayor := n2;
   end if;

   if (n3>=mayor)
     then mayor := n3;
   end if;


    RETURN mayor;  -- Devuelve el mayor de los 3 argumentos

END MAYOR_DE_3;
/
```

Si ahora quisiéramos invocar desde la consola a la función **MAYOR_DE_3**, podríamos hacer:

```
BEGIN
 DBMS_OUTPUT.PUT_LINE (MAYOR_DE_3 (11, 17, 13));
END;
/
```

Al igual que los procedimientos, las funciones pueden ser integradas en los paquetes.

CIFP CARLOS III
C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

AYUDA 5:

Tal y como se expone en el contenido de la UT6, dentro de los procedimientos y funciones podemos utilizar consultas SQL, cursores, estructuras de control (Condicionales, bucles....etc).

Así por ejemplo, como tenemos a nuestra disposición las tablas discográficas, vamos a construir una función DAME_MUSICO que reciba el código de un músico (*cod_musico*) y nos devuelva el nombre del mismo como VARCHAR2.

```
CREATE or REPLACE FUNCTION DAME_MUSICO (musico NUMBER)
 RETURN VARCHAR2

AS -- Declaraciones de variables
valor MUSICOS.nombre_musico%TYPE;

BEGIN


 SELECT nombre_musico INTO valor
 FROM MUSICOS M
 WHERE M.cod_musico = musico;


 RETURN valor;

EXCEPTION
 WHEN NO_DATA_FOUND THEN
  RAISE_APPLICATION_ERROR (-20000, 'Musico desconocido');

 END DAME_MUSICO;
/
```

CIFP CARLOS III
C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

Si quisiéramos invocar a la función **DAME_MUSICO** con el cod_musico=20, haríamos:

```
BEGIN
 DBMS_OUTPUT.PUT_LINE (DAME_MUSICO (20));
END;
/
```

PSEUDO-AYUDA 6:

En el contenido de la UT6 se han examinado cursores implícitos y explícitos, junto con toda una serie de elementos del programación, estructuras de control, manejo de excepciones...etc. Todos esos elementos serán necesarios para la realización de esta tarea.

Criterios de puntuación. Total 10 puntos.

- ✓ Construcción correcta del paquete (Vacío, sin procedimientos): **1** punto.
- ✓ Construcción correcta del procedimiento **1**: **1.5** puntos.
- ✓ Construcción correcta del procedimiento **2**: **2** puntos.
- ✓ Construcción correcta del procedimiento **3**: **2.5** puntos.
- ✓ Construcción correcta del disparador **4**: **3** puntos.

OJO: Se puede entregar el paquete sin procedimientos o con sólo alguno ó algunos de ellos. Es decir, si se ha conseguido desarrollar sólo 2 de los procedimientos, se puede entregar el paquete con sólo esos 2 procedimientos.

CIFP CARLOS III
C/Carlos III, 3
30201 CARTAGENA
TEL: 968 321301
FAX: 968 320111
30010930@educarm.es
www.cifpcarlos3.es

Indicaciones de entrega.

La tarea se entrega con un único archivo de **texto plano** que contenga la declaración y la construcción del paquete ***Gestion_Discografica*** (Conteniendo alguno, algunos ó los 3 procedimientos) y el disparador.

El archivo ha de ser de texto plano (Notepad, Edit, Notepad+...etc). Para la corrección, se copiarán/pegarán en la consola de Oracle la declaración y la construcción del paquete. Si se produce cualquier tipo de error sintáctico que impida la creación del paquete, la tarea se considerará NULA.

A continuación, si el paquete se ha creado con éxito, se probarán uno por uno los 3 procedimientos y el disparador para comprobar su funcionamiento.

El nombre que hay darle al archivo es:

apellido1_apellido2_nombre_BD_Tarea06.txt

Asegúrate de que el nombre no contenga la letra ñ, tildes ni caracteres especiales extraños. Así por ejemplo el alumno Baltasar Tenazo Fuerte, debería nombrar los archivos como:

Tenazo _Fuerte_Baltasar_BD_Tarea06.txt

Fecha límite de entrega:

Domingo 10 de Mayo de 2020, 23:55 horas.