

~~X~~ Egibide

89  
100

DATU-BASEAK  
BASE DE DATOS

ABIZENAK/APELLIDOS: Puelles López  
IZENA/NOMBRE: Unna

DATA/FECHA: 28/05/2011  
TALDEA/GRUPO: 16-DAW

USUARIO S.O.: Agbclera02 USUARIO B.D.: era02 PUESTO DEL AULA: P02

Si no existe, dentro de la carpeta documentos del usuario con el que te has conectado, una carpeta llamada 1GBD, créala. En esa carpeta guardarás la solución a los ejercicios planteados. Escribe las soluciones a cada uno de los ejercicios planteados en el fichero cuyo nombre que siga el patrón:

`yy_nombre_primer_apellido_reto.sql`

donde yy es el puesto en el que estas sentado. Recuerda poner tu nombre dentro el documento.

### PARTE 1 -120minutos-

Descárgate el script, que hay en ikas para crear las tablas , denominado tablas\_esport.sql

**Para superar esta parte no se puede tener 0 en ninguna de las subpartes.**

#### Subparte1

- 17
1. Crea un procedimiento anónimo para probar el procedimiento almacenado que encontrarás en ikas dentro del fichero ResulPartidosPorJornada.sql.

Haz la prueba para la jornada 1.

**En la solución debe de aparecer el resultado de la ejecución del procedimiento anónimo.(20 puntos/100 puntos)**

- 82
2. Modifica el procedimiento almacenado ResulPartidosPorJornada anterior, para que se denomine ResulPartidosPorJornada\_YY y devuelva, además de los datos actuales, los nombres de los equipos local y visitante, así como los puntos obtenidos en cada partido jugado.(15 puntos/100 puntos)

#### Subparte2

- 50
3. Codifica un trigger llamado maximo\_num\_jugador\_yy. donde yy es el puesto en el que estas sentado.

El trigger debe garantizar que un equipo no puede tener mas de 5 jugadores. Tener en cuenta que un jugador puede cambiar de equipo durante la temporada.

Prueba el Trigger.

- cambiando el jugador jug15 del equipo 2 al equipo 1. Adjunta pantallazo del resultado.
- Insertando un jugador nuevo. Adjunta pantallazo del resultado.

(50 puntos/100 puntos)

### PARTE 2 -15 minutos- (15 puntos/100 puntos)

Realiza el test que encontrarás en Socrative.

IMPORTANTE: Las respuestas incorrectas descuentan, si no deseas responder a una pregunta NO marques ninguna opción. Una vez marcada una opción ya no podrás dejar sin responder esa pregunta.



```

1  /*Author Unai Puelles*/
2
3  /*Subparte 1*/
4
5  --1
6
7  DECLARE
8      cursor_out SYS_REFCURSOR;
9      v_cod partidos.cod%TYPE;
10     v_fecha partidos.fecha%TYPE;
11     v_equipo_local partidos.codequipo_local%TYPE;
12     v_equipo_visitante partidos.codequipo_visitante%TYPE;
13 BEGIN
14
15     ResulPartidosPorJornada(1, cursor_out);
16
17     LOOP
18     EXIT WHEN cursor_out%NOTFOUND;
19     FETCH cursor_out INTO v_cod, v_fecha, v_equipo_local, v_equipo_visitante;
20     DBMS_OUTPUT.PUT_LINE('Cod:' || v_cod || ' Fecha:' || v_fecha || '
21     'Código equipo local:' || v_equipo_local || ' Código equipo visitante:' ||
22     v_equipo_visitante);
23     END LOOP;
24     END;
25
26     Procedimiento PL/SQL terminado correctamente.
27     Cod:1 Fecha:03/05/18 Código equipo local:1 Código equipo visitante:2
28     Cod:2 Fecha:07/05/18 Código equipo local:3 Código equipo visitante:4
29     Cod:2 Fecha:07/05/18 Código equipo local:3 Código equipo visitante:4
30     */
31     --2
32
33     CREATE OR REPLACE
34     PROCEDURE ResulPartidosPorJornada_02 (p_cod_jor integer, C_partidos OUT
35     SYS_REFCURSOR) AS
36     BEGIN
37     -- Abrir el cursor y llenarlo con datos
38     OPEN C_partidos FOR
39     SELECT P.cod, P.fecha,
40     P.codEquipo_Local, l.nombre, P.codEquipo_Visitante, p.resultadoel,
41     p.resultadoev
42     FROM jornadas J, partidos P, equipos l, equipos v
43     WHERE P.jornada_cod = J.cod
44     AND J.cod=p_cod_jor
45     AND P.jugado='S'
46     AND l.cod = P.CODEQUIPO_LOCAL OR v.cod = P.CODEQUIPO_VISITANTE
47     ;
48     END;
49
50     /*Subparte 2*/
51
52     --3
53     SET SERVEROUTPUT ON;
54
55     CREATE OR REPLACE PACKAGE trigg_pack IS
56     temp_jugador jugadores%ROWTYPE;
57     END;
58     /
59     CREATE OR REPLACE TRIGGER maximo_num_jugador_02_t
60     AFTER INSERT OR UPDATE ON jugadores
61     DECLARE
62     --Declaramos la variable en la que guardaremos el numero de jugadores del
63     equipo y la excepcion personalizada
64     v_jug_count VARCHAR2(2);
65     max_jug_err EXCEPTION;
66 BEGIN
67     --Seleccionamos el número de jugadores que está en el equipo
68     SELECT count(*) INTO v_jug_count
69     FROM jugadores

```

*¿cabeza? y control de kble main?*

*Talk el nombre del equipo visitante*



```

70 WHERE equipo_cod = trigg_pack.temp_jugador.equipo_cod;
71 --Miramos si es mayor que 5
72 IF(v_jug_count >= 5)
73 THEN
74 --Si es mayor que 5 lanzamos la excepcion max_jug_err
75 RAISE max_jug_err;
76 END IF;
77
78 EXCEPTION
79
80 WHEN max_jug_err
81 THEN
82 --Sacaremos un mensaje de error
83 RAISE_APPLICATION_ERROR(-20000, 'No se puede hacer la accion porque el
84 equipo contiene ya 5 jugadores');
85
86 END maximo_num_jugador_02_t;
87 /
88 CREATE OR REPLACE TRIGGER maximo_num_jugador_02
89 AFTER INSERT OR UPDATE ON jugadores
90 FOR EACH ROW
91 DECLARE
92 --Declaramos la variable en la que guardaremos el numero de jugadores del
93 equipo y la excepcion personalizada
94
95 BEGIN
96 --Preguntamos si está haciendo una update
97 IF UPDATING
98 THEN
99 --Si está haciendo una update le asignamos solo el valor equipo_cod al
100 jugador temporal
101 trigg_pack.temp_jugador.equipo_cod := :new.equipo_cod;
102 ELSE
103 --Si pasa aqui significa que está insertando por lo que le asignamos todas la
104 variables posibles a la variable temp_jugador
105 trigg_pack.temp_jugador.cod := :new.cod;
106 trigg_pack.temp_jugador.nombre := :new.nombre;
107 trigg_pack.temp_jugador.apellido := :new.apellido;
108 trigg_pack.temp_jugador.nickname := :new.nickname;
109 trigg_pack.temp_jugador.sueldo := :new.sueldo;
110 trigg_pack.temp_jugador.equipo_cod := :new.equipo_cod;
111 END IF;
112 END maximo_num_jugador_02;
113
114 UPDATE jugadores
115 SET equipo_cod = 1
116 WHERE upper(nombre) = 'JUG15';
117 /*
118 Error que empieza en la línea: 103 del comando :
119 UPDATE jugadores
120 SET equipo_cod = 1
121 WHERE upper(nombre) = 'JUG15'
122 Informe de error -
123 Error SQL: ORA-20000: No se puede hacer la accion porque el equipo contiene ya
124 5 jugadores
125 ORA-06512: en "EXA02.MAXIMO_NUM_JUGADOR_02_T", línea 20
126 ORA-04088: error durante la ejecución del disparador
127 'EXA02.MAXIMO_NUM_JUGADOR_02_T'
128 20000. 00000 - "s"
129 *Cause: The stored procedure 'raise_application_error'
130 was called which causes this error to be generated.
131 *Action: Correct the problem as described in the error message or contact
132 the application administrator or DBA for more information.
133 */
134
135 INSERT INTO jugadores (nombre, nickname, equipo_cod) VALUES('jug17', 'jug17', 2);
136 /*
137 Error que empieza en la línea: 125 del comando :
138 INSERT INTO jugadores (nombre, nickname, equipo_cod) VALUES('jug17', 'jug17', 2)
139 Informe de error -
140 Error SQL: ORA-20000: No se puede hacer la accion porque el equipo contiene ya

```

5 jugadores

ORA-06512: en "EXA02.MAXIMO\_NUM\_JUGADOR\_02\_T", línea 20

ORA-04088: error durante la ejecución del disparador

'EXA02.MAXIMO\_NUM\_JUGADOR\_02\_T'

20000.00000 - "%s"

\*Cause: The stored procedure 'raise\_application\_error'

was called which causes this error to be generated.

\*Action: Correct the problem as described in the error message or contact  
the application administrator or DBA for more information.

\*/



prueba\_teoría\_PL/SQL\_171

 14  
 93% (28/30)

- ✓ 1. El motor PL / SQL ejecuta los comandos de procedimiento y pasa los comandos SQL para que el servidor Oracle los procese.
- ☐ A True
- ☐ B False
- ✓ 2. PL / SQL admite sentencias CREATE
- ☐ A True
- ☐ B False
- ✓ 3. Las rutinas escritas en PL / SQL se pueden llamar en la interfaz de llamada de Oracle, Java, Pro \* C / C ++, COBOL, etc.
- ☐ A True
- ☐ B False
- ✓ 4. ¿Cuál de las siguientes afirmaciones es verdadera sobre la sección de ejecución de un bloque PL / SQL?
- ☐ A Está encerrada entre las palabras clave BEGIN y END.
- ☐ B Es una sección obligatoria.
- ☐ C Consiste en las instrucciones ejecutables PL / SQL.
- ☐ D Todo lo anterior es cierto.
- ✓ 5. Las palabras clave DECLARE, BEGIN y EXCEPTION van seguidas de punto y coma.
- ☐ A True
- ☐ B False
- ✓ 6. En la estructura básica de un bloque PL/SQL, la declaración de constantes y variables se realiza dentro de ... (indica el nombre de la sección)
- DECLARE
- ✓ 7. Teniendo en cuenta la imagen, ¿qué tipo de dato contendrá la variable DNI2?
- ☐ A Alfanumérico
- ☐ B Del mismo tipo que la columna DNI en la tabla
- ☐ C Del mismo tipo que la variable DNI
- ☐ D Ninguna es correcta

```

DECLARE
DNI1 NUMBER(4,1);
--
--
--
CONSTANT DNI2 VARCHAR2(10);
DNI2 DNI1%TYPE;

```



- ✓ 8. ¿Sería correcta la siguiente declaración de asignación?  
saldo = saldo + 2000;
- ☐ A True
- ☒ B False
- ✓ 9. ¿Con qué procedimiento puedo realizar una Salida de un bloque PL/SQL?
- ☐ A DBMS\_OUTPUT.PUT\_LINE
- ☐ B DMS\_OUPUT.PUT\_LINE
- ☐ C DBM\_OUPUT.PUT\_LINE
- ☒ D DBMS\_OUTPUT.PUT\_LINE
- ✓ 10. Para obtener el resultado de salida del servidor y mostrarlo en la pantalla, debe escribir:
- ☐ A set serveroutput on
- ☐ B set server output on
- ☐ C set dbmsoutput on
- ☐ D set dbms output on
- ✓ 11. En un bloque PL/SQL podemos usar excepciones para tratar errores. ¿con cada excepción podremos tratar un sólo error o mas de uno? Responde: mas o uno.
- UNO
- ✓ 12. En un bloque PL/SQL podemos usar excepciones para tratar errores. Oracle ya tiene predefinido un conjunto de excepciones. Si el error que deseamos controlar no está en ese conjunto no podemos hacer nada.
- ☐ A True
- ☒ B False

✓ 13. ¿Cuál de las siguientes afirmaciones es verdadera sobre el siguiente fragmento de código?

```
DECLARE
a number(3) := 100;
BEGIN
IF ( a = 50 ) THEN
dbms_output.put_line('El valor exacto de a es: 10' );
ELSIF ( a = 75 ) THEN
dbms_output.put_line('El valor exacto de a es: 20' );
ELSE
dbms_output.put_line('Ninguno de los valores coincide');
END IF;
dbms_output.put_line('El valor exacto de a es: ' || a );
END;
```

- ☐ A Tiene un error de sintaxis.
- ☐ B Imprimirá 'Ninguno de los valores coincide'.
- ☐ C Se imprimirá:  
Ninguno de los valores coincide
- ☐ D Ninguna de las anteriores.

El valor exacto de a es: 100

✓ 14. ¿Qué devolverá el fragmento de código que ves en la imagen?

- ☐ A Very good
- ☐ B No such grade
- ☐ C B
- ☐ D Hay un error de sintaxis, no devolverá nada.

```
DECLARE
grade char(1) := 'B';
BEGIN
CASE
when grade = 'A' then dbms_output.put_line('Excellent');
when grade = 'B' then dbms_output.put_line('Very good');
when grade = 'C' then dbms_output.put_line('Well done');
when grade = 'D' then dbms_output.put_line('You passed');
when grade = 'F' then dbms_output.put_line('Better try again');
else dbms_output.put_line('No such grade');
end case;
END;
```

✓ 15. ¿Cuál es la salida del código que ves en la imagen?

- ☐ A 6  
4
- ☐ B 4  
5
- ☐ C 5  
4
- ☐ D Ninguna es correcta

```
DECLARE
x number := 4;
BEGIN
LOOP
dbms_output.put_line(x);
x := x + 1;
exit WHEN x > 5;
END LOOP;
dbms_output.put_line(x);
END;
```



✗ 16. ¿Cuál es la salida del código de la imagen?

- ☐ A 1
- ☐ B Hay un error sintáctico, no devolverá nada
- ☐ C 0
- ☐ D Ninguna es correcta

```
DECLARE
  mynum pls_integer := 0;
BEGIN
  LOOP
    mynum := mynum + 1;
    CASE mynum
      WHEN 1 THEN dbms_output.put_line('1');
      WHEN 2 THEN dbms_output.put_line('2');
      WHEN 3 THEN dbms_output.put_line('3');
    END CASE;
    EXIT;
  END LOOP;
END;
```

✓ 17. ¿Qué hace el siguiente código PL/SQL:  
INVALID\_DATE EXCEPTION;

- ☐ A Manejar una excepción
- ☒ B Declarar una excepción
- ☐ C Una excepción es asociada
- ☐ D Ninguna es cierta

✓ 18. ¿Qué tipo de excepción requiere el uso de la instrucción RAISE?

- ☐ A Una excepción de servidor sin nombre
- ☒ B Una excepción definida por el programador
- ☐ C Una excepción de servidor con nombre
- ☐ D La sentencia RAISE nunca hace uso de una excepción.

✓ 19. ¿Cuándo se ejecutará el código PL/SQL asociado a la excepción predefinida  
TOO\_MANY\_ROWS?

- ☐ A A un objeto nulo se le asigna automáticamente un valor
- ☒ B Una sentencia SELECT INTO devuelve mas de una fila
- ☐ C Un cursor obtiene valor en una variable que tiene un tipo de datos incompatible
- ☐ D PL / SQL se quedó sin memoria o la memoria se dañó

✓ 20. La excepción predefinida NO\_DATA\_FOUND es ejecutada cuando ...

- ☐ A A un objeto nulo se le asigna automáticamente un valor.
- ☐ B Una sentencia SELECT INTO devuelve mas de una fila.
- ☐ C Un cursor obtiene valor en una variable que tiene un tipo de datos incompatible.
- ☒ D Una sentencia SELECT INTO no devuelve ninguna fila.



21. ¿Cómo se hace la asignación de un código del servidor Oracle a la excepción definida por el usuario?

- ☐ (A) user\_code exception
- ☐ (B) pragma exception\_user\_code
- ☒ (C) pragma exception\_init
- ☐ (D) pragma init\_exception

✓ 22. Considere la excepción declarada como:  
emp\_exc1 EXCEPTION;

¿Cuál de las siguientes afirmaciones llamará correctamente a la excepción en un bloque PL / SQL?

- ☒ (A) IF c\_id <=0 THEN RAISE emp\_exc1;
- ☐ (B) IF c\_id <=0 THEN CALL emp\_exc1;
- ☐ (C) IF c\_id <=0 THEN EXCEPTION emp\_exc1;
- ☐ (D) IF c\_id <=0 THEN emp\_exc1;

✓ 23. ¿Cuál de las siguientes sintaxis es correcta para crear un cursor explícito?

- ☐ (A) CURSOR nom\_cursor IS sentencia-select;
- ☐ (B) CREATE CURSOR nom\_cursor AS sentencia-select;
- ☐ (C) CURSOR nom\_cursor AS sentencia-select;
- ☐ (D) CREATE CURSOR nom\_cursor IS sentencia-select;

✓ 24. ¿Qué atributo de cursor (junto con el nombre del cursor) deberías utilizar para programar la salida de un bucle LOOP en la siguiente situación:  
"Si la sentencia FETCH no devuelve ningún registro para el cursor debemos de salir."

- ☐ (A) cur\_emp%NOTCOUNT
- ☒ (B) cur\_emp%NOTFOUND
- ☐ (C) cur\_emp%FOUND
- ☐ (D) cur\_emp%COUNT

✓ 25. ¿Cuál de las siguientes afirmaciones es cierta sobre los cursores en PL/SQL?

- ☐ (A) Los cursores implícitos son cursores creados automáticamente por Oracle.
- ☐ (B) El cursor implícito tiene de nombre SQL y posee los atributos como % FOUND,% ISOPEN,% NOTFOUND y% ROWCOUNT
- ☐ (C) Los cursores explícitos son definidos por el programador.
- ☒ (D) Todas son ciertas



✓ 26. Observa el siguiente código y rellena los huecos:

- (A) %FOUND, %NOTFOUND, %ROWCOUNT
- (B) SQL%FOUND, SQL%NOTFOUND, SQL%ROWCOUNT
- (C) %NOTFOUND, %FOUND, %ROWCOUNT
- (D) SQL%NOTFOUND, SQL%FOUND, SQL%ROWCOUNT

```
DECLARE
  total_rows number(3);
BEGIN
  UPDATE employees
  SET salary = salary + 500;
  IF
    THEN
  dbms_output.put_line('No se actualizaron empleados');
  ELSE
    THEN
  total_rows :=
  dbms_output.put_line('Total rows to be updated: ' || total_rows);
END IF;
```

✓ 27. Indica por qué falla el siguiente código:

```
DECLARE
  V_BONUS number;
BEGIN
  SELECT salario INTO V_SALARY
  FROM emple
  WHERE EMP_NO=101;
  V_BONUS:=V_SALARY*0.1;
END;
```

- (A) Falta la sección EXCEPTION
- (B) V\_BONUS es una variable no inicializada.
- (C) Ninguna de ellas.
- (D) V\_SALARY es una variable no declarada.

✓ 28. Según el siguiente código, indica cuando se alcanzará la excepción programada:

```
DECLARE
  V_SALARY number;
BEGIN
  SELECT SUM(salario) INTO V_SALARY
  FROM emple;
  dbms_output.put_line('El coste en salarios para la compañía es: ' ||
    '|| TO_CHAR(V_SALARY));
  EXCEPTION
  WHEN NO_DATA_FOUND THEN
  dbms_output.put_line('Ningún valor');
END;
```

- (A) Cuando la tabla emple este vacía.
- (B) No se ejecutará pues hay un error sintáctico.
- (C) La sentencia SELECT no puede usar funciones de grupo.
- (D) Nunca.