EXAMEN DE JUNIO DE BASES DE DATOS. 16/05/2022

DML.PLSQL

NOMBRE:_____

La empresa CLINICSALUD formada por un conjunto de clínicas privada. Se desea informatizar el sistema de visitas de los pacientes con los distintos médicos que trabajan en las distintas clínicas. Tras el estudio se han obtenido las siguientes tablas y campos:

- **CLÍNICA: Código,** nombre, dirección, localidad, provincia
- **MEDICO: N_Colegiado,** Nombre, Apellidos, Especialidad, Fecha_alta, clínica
- **CONSULTAS: Médico, Paciente, Fecha_consulta,** Importe
- **PACIENTE: NSeguridadSocial**, DNI, Nombre, Apellidos, Sexo, Fecha de nacimiento, Total_gastado

EJERCICIOS DML. (3p)

Realiza las siguientes operaciones de actualización, inserción y borrado de registros mediante las sentencias SQL apropiadas:

1. Inserta un nuevo paciente con NSeguridadSocial='156487569', DNI='26485695K', Nombre= 'Juan', Apellidos= 'López Alcántara, Fec_nac='28/10/1986' y Sexo='v'. Crea otra sentencia que almacene en la tabla CONSULTAS una consulta para este nuevo paciente con el médico con N_colegiado='1586865' para el 10/05/2023 con un importe de 50 euros (0,3p)

2. Incrementar en 10 euros el importe de las consultas que estén por debajo de la media de los importes de todas las consultas. (0,5p)

3. Realiza un descuento de 20 euros en el importe de todas las consultas de los pacientes de Madrid que hayan asistido a consulta más de 10 veces. (0,6p)

4. Bloquea la tabla CONSULTAS en modo lectura seguidamente actualiza el importe de las consultas realizadas por médicos de la especialidad de 'Traumatología' incrementándolas en un 10%. (0,5p)

5. Inicia una transacción. Elimina todos los pacientes que lleven más de 2 años sin asistir a consulta médica. Deshacer la transacción. (0,6p)

EJERCICIOS PL-SQL. (7 p)

Utilizando las mismas tablas, realiza con el lenguaje PL/SQL los siguientes apartados:

1. Crear un procedimiento que reciba como parámetro el Número de colegiado de un médico y muestre por pantalla el nombre, apellidos y especialidad del médico así como un listado con el NSeguridadSocial, DNI, nombre y apellidos, fecha de consulta e importe de los pacientes que hayan asistido a consulta con ese médico. En la última línea debe aparecer la suma total de los importes obtenidos por el médico en las consultas. Realiza el control de excepciones para el caso de que el médico pasado como parámetro no exista y para el caso de que el médico no haya realizado ninguna consulta. (3p)

2. Crea una función que reciba como parámetro el nombre y apellidos de un paciente y devuelva el tipo de paciente que será:

- ‘ Bronce’ si el número de consultas a las que ha asistido es menor o igual a 5.

- ‘Plata’ si el número de consultas está entre 5 y 10.

- ‘Oro’ si ha asistido a más de 10 consultas.

Realiza el control de errores para el caso de que el paciente no exista y el bloque de sentencias necesario para probar la función con pacientes que existan y que no existan. (2p)

3. Crear un disparador (trigger) que cada vez que se inserte o actualice un registro de la tabla CONSULTAS con un valor de importe distinto de NULL se actualice el campo ‘total_gastado’ de la tabla PACIENTE sumando a este campo el valor del campo ‘importe’. (2p)

ANEXO: Estructuras PL/SQL que pueden ser útiles para la realización del examen práctico.

Procedimiento en PL/SQL

```
CREATE[OR REPLACE] PROCEDURE nombre_procedimiento([parámetros])
IS
[DECLARE]
        [<variables locales>]
BEGIN
        <código del procedimiento>
[EXCEPTION]
END [nombre_procedimiento];
```

Función en PL/SQL

```
CREATE [OR REPLACE] FUNCTION nombre_función([parámetros])
RETURN tipodato
IS
[DECLARE]
        [<variables locales>]
BEGIN
        <código de la función>
RETURN <valor>;
[EXCEPTION]
END [nombre_función];
```

Disparador en PL/SQL

```
CREATE[OR REPLACE]TRIGGER nombre_trigger
{BEFORE|AFTER|INSTEAD OF}
{INSERT|DELETE|UPDATE [OF <atributo]} ON <tabla>
[FOR EACH ROW|STATEMENT]
[WHEN condición]
[DECLARE]
        [<variables locales>]
BEGIN
        <código del trigger>
[EXCEPTION]
END[nombre_trigger];
```

Declaración y manejo de un cursor en PL/SQL

CURSOR prueba **IS** Sentencia de consulta;	
FETCH prueba **INTO** mi_registro; **WHILE** prueba%FOUND **LOOP** DBMS_OUTPUT.PUT_LINE('Hola'); **FETCH** prueba **INTO** mi_registro; **END LOOP**;	

Estructura de control en PL/SQL

```
IF condicion THEN
DBMS_OUTPUT.PUT_LINE('lo que sea');
ELSE
DBMS_OUTPUT.PUT_LINE(una_variable);
END IF;
```