Para la realización de los ejercicios de esta unidad nos basaremos en el caso de estudio expuesto en los contenidos de la misma en el Anexo I. La tarea que te pedimos que realices consta de 2 actividades:

Ejercicio 1

Teniendo en cuenta que en el caso de estudio una de las tablas es FAMILIA, que su estructura es (identificador(pk), nombre, familia(fk), oficina(fk)), que contiene información acerca de una familia, su nombre, el identificador de la familia de la que depende (su madre) y su oficina.

Queremos crear un procedimiento que tendrá la siguiente cabecera mover_familia (id_origen, id_destino), donde cada uno de los argumentos corresponde a un identificador de familia. El procedimiento intentará que la familia a la que corresponde id_origen, pase a ser la hija de la familia identificada por id_destino, o de otra manera, su familia madre será la especificada en el argumento id_destino.

Para realizar esta operación, será necesario comprobar que ambas familias existen y que no son iguales.

Para la comprobación de la existencia de dichas familias se puede utilizar un cursor variable y en caso de que no exista, se visualizará el mensaje correspondiente mediante una excepción del tipo RAISE_APPLICATION_ERROR. También se mostrará un mensaje en caso de que ambos argumentos tengan el mismo valor.

Además si la familia origen pertenecía a una oficina deberá dejar de pertenecer a esa oficina, ya no pertenecerá a ninguna y sólo ser hija de la familia destino.

SOLUCION

```
CREATE OR REPLACE PROCEDURE mover_familia(id_origen NUMBER, id_destino NUMBER) IS
     familia_origen familias%ROWTYPE;
     familia_destino familias%ROWTYPE;
     TYPE cursor_familias IS REF CURSOR RETURN familias%ROWTYPE;
     cFamilias cursor_familias;

     --Función auxiliar que nos devuelve 0 si una familia no es hija de otra y 1 en caso contrario
     FUNCTION es_hija(origen familias%ROWTYPE, destino familias%ROWTYPE) RETURN NUMBER IS
          madre familias%ROWTYPE;
     BEGIN
          IF (destino.familia IS NULL) THEN
               RETURN 0;
          ELSIF (destino.familia = origen.identificador) THEN
               RETURN 1;
          ELSE
               SELECT * INTO madre FROM familias WHERE identificador = destino.familia;
               RETURN es_hija(origen, madre);
          END IF;
     END;
```

```
BEGIN
     --Comprobamos si la familia origen existe y si existe la guardamos en familia_origen
     OPEN cFamilias FOR SELECT * FROM familias WHERE identificador = id_origen;
     FETCH cFamilias INTO familia_origen;
     IF (cFamilias%FOUND = FALSE) THEN
          RAISE_APPLICATION_ERROR(-20011, 'La familia origen no existe');
     END IF;

     --Comprobamos si la familia destino existe y si existe la guardamos en familia_destino
     OPEN cFamilias FOR SELECT * FROM familias WHERE identificador = id_destino;
     FETCH cFamilias INTO familia_destino;
     IF (cFamilias%FOUND = FALSE) THEN
          RAISE_APPLICATION_ERROR(-20012, 'La familia destino no existe');
     END IF;

     --Comprobamos si la familia destino es hija de la familia origen
     IF (es_hija(familia_origen, familia_destino) = 1) THEN
          RAISE APPLICATION ERROR(-20013, 'La familia destino es hija (directa o indirecta) de la familia
origen');
     END IF;

     --Actualizamos la familia al identificador de la familia destino
     --y ponemos la oficina a NULL por si acaso era la familia raíz de la oficina
     UPDATE familias SET familia = id_destino, oficina = NULL WHERE identificador = id_origen;

     COMMIT;
END;
/
```

Ejercicio 2.

Queremos controlar algunas restricciones a la hora de trabajar con agentes:

- ✓ El usuario y la clave de un agente no pueden ser iguales.
- ✓ La habilidad de un agente debe estar comprendida entre 0 y 9 (ambos inclusive).
- ✓ La categoría de un agente sólo puede ser igual a 0, 1 o 2.
- ✓ Si un agente pertenece a una oficina directamente, su categoría debe ser igual 2.
- ✓ Si un agente no pertenece a una oficina directamente, su categoría no puede ser 2.
- ✓ No puede haber agentes que no pertenezcan a una oficina o a una familia.
- ✓ No puede haber agentes que pertenezcan a una oficina y a una familia a la vez.

Debes crear un disparador para asegurar estas restricciones. El disparador deberá lanzar todos los errores que se puedan producir en su ejecución mediante errores que identifiquen con un mensaje adecuado por qué se ha producido dicho error.

SOLUCION

```
CREATE OR REPLACE TRIGGER integridad_agentes
BEFORE INSERT OR UPDATE ON agentes
FOR EACH ROW
BEGIN
     --Comprobamos que el usuario y la clave no son iguales
     IF (:new.usuario = :new.clave) THEN
          RAISE_APPLICATION_ERROR(-20021, 'El usuario y la clave deben ser diferentes');
     END IF;

     --Comprobamos que la habilidad del agente está comprendida entre 0 y 9
     IF (:new.habilidad < 0 OR :new.habilidad > 9) THEN
          RAISE_APPLICATION_ERROR(-20022, 'La habilidad del agente es errónea');
     END IF;

     --Comprobamos que la categoria del agente está comprendida entre 0 y 2
     IF (:new.categoria < 0 OR :new.categoria > 2) THEN
          RAISE_APPLICATION_ERROR(-20023, 'La categoría del agente es errónea');
     END IF;

     --Si un agente pertenece directamente a una oficina su categoria debe ser 2
     IF (:new.oficina IS NOT NULL and :new.categoria != 2) THEN
          RAISE_APPLICATION_ERROR(-20024, 'La categoría de un agente que pertenece a una oficina directamente debe ser
2');
     END IF;

     --Si un agente no pertenece directamente a una oficina su categoria debe ser distinta de 2
     IF (:new.oficina IS NULL and :new.categoria = 2) THEN
          RAISE_APPLICATION_ERROR(-20025, 'La categoría de un agente que no pertenece a una oficina directamente debe
ser distinta de 2');
     END IF;

     --No puede haber agentes huérfanos ni con dos padres (oficina y familia)
     IF (:new.familia IS NULL and :new.oficina IS NULL) THEN
          RAISE_APPLICATION_ERROR(-20026, 'Un agente no puede ser huérfano');
     ELSIF (:new.familia IS NOT NULL and :new.oficina IS NOT NULL) THEN
          RAISE_APPLICATION_ERROR(-20027, 'Un agente no puede tener dos padres');
     END IF;
END;
/
```