



ETSI Informática (SISTEMAS) BASES DE DATOS Relación de Problemas VII PL/SQL

1. Cierta usuario (o grupo de usuarios) queremos que no pueda insertar un empleado en la tabla de EMPLEADOS si no existe otro empleado en la empresa que trabaja para otro departamento y cobra la misma comisión que él. Nota: utilice un cursor para buscar el segundo empleado y comprobar su existencia.

Para controlar esta restricción se ha decidido quitarle el permiso para insertar (INSERT) en la tabla EMPLEADOS, de forma que cuando ese usuario quiera insertar un empleado en esa tabla utilice un **procedimiento PL/SQL** que debe programarse. El procedimiento recibirá como parámetros de entrada los atributos del nuevo empleado y devolverá por pantalla la información sobre si la inserción se ha realizado o no. En caso negativo el mensaje de error al usuario será del tipo: "No es posible realizar la inserción del empleado <CODIGO> porque no hay otro empleado que trabaje en otro departamento que no sea <NOMBRE_DPTO> y que tenga como comisión <VALOR_COMISION>".

2. Realice un bloque de **PL/SQL** (o procedimiento sin argumentos) que al ser ejecutado muestre en pantalla la información sobre los objetos creados por el usuario. Para ello, el bloque debe recorrer las tablas USER_TABLES, USER_CONSTRAINTS y USER_VIEWS para luego mostrar la información del siguiente modo:

==> Se han creado X vistas de los siguientes tipos:

- NOMBRE_VISTA_1: cuya sentencia de definición es SENTENCIA_SELECT
- NOMBRE_VISTA_2: cuya sentencia de definición es SENTENCIA_SELECT
- ...

==> Existe una tabla llamada NOMBRE_TABLA_1 que tiene actualmente X_1 filas y X_2 restricciones de los siguientes tipos:

- NOMBRE_REST_1: Tipo PRIMARY KEY/UNIQUE/NOT NULL/CHECK.
- NOMBRE_REST_2: Tipo PRIMARY KEY/UNIQUE/NOT NULL/CHECK.
- ...

Nota: Se puede mejorar este bloque PL/SQL visualizando también los datos de los procedimientos, funciones, paquetes y disparadores (*triggers*). Para ello usar las vistas del diccionario de datos de Oracle como USER_SOURCE y USER_TRIGGERS.

3. Se quiere obtener el valor de los códigos de departamentos que tienen el mayor número de empleados. Esto se puede realizar con la siguiente consulta SQL:

```
SELECT DISTINCT Departamento
FROM Empleados
GROUP BY Departamento
HAVING COUNT(*) >= ALL (SELECT COUNT(*)
                        FROM Empleados
                        GROUP BY Departamento);
```

Dicha consulta tiene un coste bastante elevado y, por ello se propone realizarla de dos formas distintas creando dos bloques **PL/SQL** que cubren los siguientes razonamientos:

- ♦ Creando una tabla: Crear una **tabla** llamada INFO_DPTOS que contenga los valores de código de los departamentos y número de empleados en cada uno (sacar estos datos de la tabla de empleados). A continuación se puede obtener el valor máximo de número de empleados (consultando la tabla INFO_DPTOS) y recorrer la misma tabla para encontrar qué departamentos coinciden con ese valor de número de empleados.
- ♦ Sin crear ninguna tabla: Se pueden **anidar dos cursores**. El primero recorre la tabla EMPLEADOS agrupada por el valor de "departamento" y busca los valores de número de empleados (N_EMP) y código del departamento (COD_DPTO). A continuación se abre un segundo cursor que busca una tupla en EMPLEADOS agrupada por "departamento", que tenga un valor mayor de número de empleados que el que está en N_EMP. Si este cursor tiene alguna tupla, entonces el departamento COD_DPTO no tiene el valor máximo de número de empleados por lo que se rechaza y seguimos buscando (se hace un nuevo FETCH del primer cursor). En caso contrario se selecciona ese departamento y seguimos buscando.