Tarefa 1. Crear e executar procedementos almacenados

A tarefa consiste en escribir os guións de sentenzas SQL necesarios para crear procedementos almacenados atendendo a varios supostos, documentando os guións, e executando os procedementos creados.

* Tarefa 1.1. Crear un procedemento almacenado na base de datos *traballadores* que actualice a columna *depEmpregados* da táboa *departamento*, para todos os departamentos, contando o número de empregados que traballan nese departamento tendo en conta a información da columna *empDepartamento* da táboa *empregado*.
* Tarefa 1.2. Crear un procedemento almacenado co nome *vertaboas* na base de datos *utilidades*, que utilice a información contida na base de datos *information\_schema* para mostrar información das táboas que hai nas bases de datos. O procedemento recibe como parámetro de entrada unha cadea e texto que pode ser o nome dunha base de datos, ou ben, o carácter asterisco ('\*').
* Cando se lle pasa o carácter '\*' debe mostrar todas as táboas do servidor. Para cada táboa nos interesan as columnas: *table\_schema, table\_name, table\_type, engine, table\_rows* da táboa *information\_schema .tables*, ordenando o resultado polo nome do esquema e o nome da táboa.
* Cando se lle pasa o nome dunha base de datos, hai que comprobar que a base de datos existe na táboa *schemata*. No caso de existir, se mostrarán todas as táboas desa base de datos. Para cada táboa nos interesan as columnas: *table\_name, table\_type, engine, table\_rows* da táboa *information\_schema .tables*, ordenando o resultado polo nome da táboa. No caso de non existir a base de datos, se mostrará unha mensaxe de erro: ' A base de datos *xxxxx* non existe no servidor '.
* Tarefa 1.3. Crear un procedemento almacenado que nos permita inserir datos de proba na táboa *vendas* na base de datos *tendaBD*.
* O número de filas a inserir se lle pasa como un parámetro.
* En cada fila, os datos para as columnas *ven\_cliente*, *ven\_tenda* e *ven\_empregado* obtéñense buscando unha fila de maneira aleatoria nas táboas *clientes*, *tendas* e *empregados* respectivamente e collendo o código que corresponde.
* A columna *ven\_data* colle a data do sistema.
* Nas columnas *ven\_id* e *ven\_factura* non se cargan datos. Na primeira porque é de tipo autoincremental e xa a calcula o servidor, e a segunda porque non se cubre ata que se facture a venda.
* Tarefa 1.4. Crear un procedemento almacenado na base de datos *tendaBD* que permita controlar os intentos de acceso errados dos usuarios da base de datos. Os parámetros de entrada son o *login* e *password* do usuario.

A táboa *usuario* garda información dos usuarios que poden acceder á base de datos, e terá o seguinte esquema:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nome columna | Tipo | Null | Clave | Observacións |
| login | varchar(16) | N | P | Nome de usuario |
| password | char(40) | N |  | Contrasinal do usuario |

A táboa de *log\_erro\_conexion* rexistra os intentos de acceso errados, e terá o seguinte esquema:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nome columna | Tipo | Null | Clave | Observacións |
| id | integer | N | P | Código autoincremental |
| login | varchar(16) | N |  | Nome de usuario |
| password | char(40) | N |  | Contrasinal do usuario |
| data\_hora | timestamp | N |  | Data e hora do intento de acceso |

O procedemento debe comprobar se existe na táboa *usuario* algún usuario co *login* e *password* que se pasan como parámetro. No caso de non existir, gárdase na táboa de rexistro *log\_erro\_conexion* a información correspondente ao intento de acceso errado. No caso de que o usuario faga máis de 5 intentos errados nos últimos 3 minutos, bloquearase a súa conta cambiándolle o contrasinal, poñendo unha contrasinal fixa establecida polo administrador, como por exemplo: 'H347B52(((]ERR'.

O procedemento utilizará un parámetro de saída para poder comprobar se o intento de acceso tivo éxito ou non. O parámetro ten o valor 0 se o *login* e o *password* corresponden a un usuario que existe na táboa; o valor 1 se o usuario non existe; o valor 2 no caso de bloqueo da conta por superar o número de intentos permitidos.

Tarefa 2. Crear e utilizar funcións definidas polo usuario

A tarefa consiste en escribir os guións de sentenzas SQL necesarios para crear funcións atendendo a varios supostos, e facer as probas de funcionamento utilizando as funcións creadas nunha consulta coa sentenza SELECT.

* Tarefa 2.1. Crear unha función na base de datos *utilidades* á que se lle pasa como parámetro o número do mes, e devolva o nome do mes en galego.
* Tarefa 2.2. Crear unha función na base de datos *utilidades* á que se lle pase como parámetro a nota numérica (dous enteiros e dous decimais) dun alumno, e devolva a nota en letra tendo en conta a seguinte táboa:

|  |  |  |
| --- | --- | --- |
| Nota numérica | | Nota en letra |
| >= 0 | < 5 | suspenso |
| >= 5 | < 6 | aprobado |
| >= 6 | < 7 | ben |
| >= 7 | < 9 | notable |
| >= 9 | <= 10 | sobresaínte |
| Outro valor | | erro na nota |

* Tarefa 2.3. Crear unha función na base de datos *utilidades* que pasándolle como parámetro as 8 cifras correspondentes ao número do DNI, devolva a letra que lle corresponde.

A letra do DNI obtense mediante un algoritmo coñecido como módulo 23. O algoritmo consiste en dividir o número correspondente ao DNI entre 23 e obter o resto da división enteira. O resultado é un número comprendido entre o 0 e o 22. A cada un destes números se lles fai corresponder unha letra tendo en conta a seguinte táboa:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
| T | R | W | A | G | M | Y | F | P | D | X | B | N | J | Z | S | Q | V | H | L | C | K | E |

Non se utilizan as letras: I, Ñ, O, e U. As letras I e O se descartan para evitar confusións con outros carácteres, como 1, l o 0.

* Tarefa 2.4. Crear unha función na base de datos *utilidades* que pasándolle como parámetro os 8 primeiros carácteres correspondentes ao Número de Identidade de Estranxeiro (NIE), devolva a letra que lle corresponde.

Utilízase o mesmo algoritmo que para o DNI, coa diferenza de que o NIE pode empezar por unha letra, polo que hai que engadirlle as seguintes restricións:

* No caso de que o NIE empece pola letra X, se calcula desprezando a X, e utilizando os 7 díxitos restantes.
* No caso de que o NIE empece pola letra Y, se substitúe a letra Y polo número 1.
* No caso de que o NIE empece pola letra Z, se substitúe a letra Z polo número 2.
* Tarefa 2.5. Crear unha función na base de datos *utilidades* á que se lle pasa como parámetro os 20 díxitos dunha conta bancaria española, e retorne como saída o IBAN que lle corresponde.

*O documento Cálculo do IBAN contén a explicación de como se calcula o IBAN para as contas bancarias de España, noutros países as contas bancarias poden ter ata 34 díxitos, e poden incluír letras.*

Tarefa 3. Modificar procedementos almacenados e funcións

A tarefa consiste en facer modificacións nos seguintes procedementos almacenados e funcións:

* Tarefa 3.1. Cambiar as seguintes características do procedemento almacenado *utilidades.vertaboas()* creado na tarefa 1.2.:
* Se teñan en conta os privilexios do usuario que o executa (INVOKER).
* Engadir como comentario o texto: 'Mostra a información das táboas das bases de datos que se pasan como parámetro'.
* Tarefa 3.2. Borrar a función letraNIE, creada na tarefa 2.4.