

Tarefa 1. Declarar variables, condicións, e manipuladores

A tarefa consiste en crear un guión coas seguintes sentenzas e comprobar con Workbench que non se cometeron erros de sintaxe:

- Declarar unha variable chamada *vNumero* que é de tipo numérica, enteira, e asignarlle o valor 0 cando se crea.
- Declarar unha condición á que se lle vai a asignar o nome *claveDuplicada* para o caso que se produza o código de erro de MySQL 1062.
- Declarar unha variable chamada *vFinal* de tipo bit e asignarlle o valor 0, e declarar un manipulador que lle asigne o valor 1 a esa variable cando se produza unha condición de erro NOT FOUND. Cando se produce o erro, debería continuar a execución do programa almacenado.

Solución

```
/*Declarar unha variable chamada vNumero que é de tipo numérica, enteira, e asignarlle o valor 0 cando se crea.*/  
declare vNumero integer default 0;  
/* Declarar unha condición á que se lle vai a asignar o nome claveDuplicada para o caso que se produza o código de erro de MySQL 1062.*/  
declare claveDuplicada condition for 1062;  
/*Declarar unha variable chamada vFinal de tipo bit e asignarlle o valor 0, e declarar un manipulador que lle asigne o valor 1 a esa variable cando se produza unha condición de erro NOT FOUND. Cando se produce o erro debería continuar a execución do programa almacenado.*/  
declare vFinal bit default 0;  
declare continue handler for not found set vFinal = 1;
```

A sentenza *declare* só se pode utilizar dentro dun bloque de programación dun programa almacenado, así que de querer ver se a sintaxe é correcta coa axuda do editor gráfico de Workbench, podemos incluílas na creación dun procedemento almacenado cun único bloque de programación:

```
delimiter //  
create procedure proba()  
begin  
/*Declarar unha variable chamada vNumero que é de tipo numérica, enteira, e asignarlle o valor 0 cando se crea.*/  
declare vNumero integer default 0;  
/* Declarar unha condición á que se lle vai a asignar o nome claveDuplicada para o caso que se produza o código de erro de MySQL 1062.*/  
declare claveDuplicada condition for 1062;  
/*Declarar unha variable chamada vFinal de tipo bit e asignarlle o valor 0, e declarar un manipulador que lle asigne o valor 1 a esa variable cando se produza unha condición de erro NOT FOUND. Cando se produce o erro debería continuar a execución do programa almacenado.*/  
declare vFinal bit default 0;  
declare continue handler for not found set vFinal = 1;  
end;  
//  
delimiter ;
```

Tarefa 2. Asignar valores a variables

A tarefa consiste en crear un gui3n coas seguintes sentenzas de asignaci3n e comprobar que funcionan:

- Asignar o valor 'servido' a unha variable local chamada *vEstado*.
- Asignar o valor 26 a unha variable de usuario chamada *idade*.
- Asignar o valor 'OFF' a unha variable de sistema chamada *autocommit*.
- Asignar a unha variable local chamada *vNomeProvincia* o contido da columna do *nome-Provincia* da provincia que ten como c3digo o valor 15, tomando os datos da t3boa *provincia* na base de datos *utilidades*.

Soluci3n

```
declare vEstado char(8);
declare vNomeProvincia varchar(35) default null;
-- Asignar o valor 'servido' a unha variable local chamada vEstado.
set vEstado = 'servido';
-- Asignar o valor 26 a unha variable de usuario chamada idade.
set @idade = 26;
select @idade := 26; # tam3n se poder3a facer con esta sentenza
-- Asignar o valor 'OFF' a unha variable de sistema chamada autocommit.
set @@autocommit = 0;
/*Asignar a unha variable local chamada vNomeProvincia o contido da columna
 nomeProvincia da provincia que ten como c3digo o valor 15, tomando os datos
 da t3boa provincia na base de datos utilidades.
*/
select nome into vNomeEmpregado
  from practicas1.empregado
 where dni = '44552010K';
```

Para probar que funcionan as ordes de asignaci3n p3dese crear un procedemento almacenado cun 3nico bloque de programaci3n:

```
set autocommit = 1;
drop procedure if exists asignarValores;
delimiter //
create procedure asignarValores()
begin
declare vEstado char(8);
declare vNomeEmpregado varchar(80) default null;
-- Asignar o valor 'servido' a unha variable local chamada vEstado.
set vEstado = 'servido';
-- Asignar o valor 26 a unha variable de usuario chamada idade.
set @idade = 26;
select @idade := 26; # tam3n se poder3a facer con esta sentenza
-- Asignar o valor 'OFF' a unha variable de sistema chamada autocommit.
set @@autocommit = 0;
/*Asignar a unha variable local chamada vNomeProvincia o contido da columna
 nomeProvincia da provincia que ten como c3digo o valor 15, tomando os datos
 da t3boa provincia na base de datos utilidades.
*/
select nome into vNomeEmpregado
  from practicas1.empregado
 where dni = '44552010K';
select vEstado,@idade,@@autocommit,vNomeEmpregado;
end;
//
delimiter ;
```

```
call asignarValores();
```

Resultado da execución:

Result Grid				Filter Rows:	Export:	Wrap Cell Content:
vEstado	@idade	@@autocommit	vNomeEmpregado			
servido	26	0	Bernardez Varela, Luis			

Tarefa 3. Utilizar a sentença if

A tarefa consiste en escribir e comprobar o funcionamento dun bloque de programación que permita mostrar o texto 'positivo' no caso de que a variable *vNumero* tome un valor maior que cero, 'negativo' en caso de que tome un valor menor a 0, e 'cero', no caso que tome o valor 0.

Solución

```
if vNumero = 0 then select 'cero';
elseif vNumero > 0 then select 'positivo';
else select 'negativo';
end if;
```

Para probar o funcionamento da sentença if, pódese crear un procedemento almacenado cun único bloque de programación:

```
/* Escribir un bloque de programación no que se mostre o texto 'positivo'
no caso de que a variable vNumero tome un valor maior que cero, 'negativo'
en caso de que tome un valor menor a 0, e 'cero', no caso que tome o valor 0.*/
drop procedure if exists tarefa3;
delimiter //
create procedure tarefa3()
begin
declare vNumero integer;
set vNumero = -90;
if vNumero = 0 then select 'cero';
elseif vNumero > 0 then select 'positivo';
else select 'negativo';
end if;
end;
//
delimiter ;
call tarefa3();
```

Resultado da execución:

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	negativo			
►	negativo			

Tarefa 4. Utilizar a sentença case

A tarefa consiste en escribir e comprobar que funcione un bloque de programación, no que se mostre a descrición do estado civil dunha persoa, en función do valor que toma a variable

vCodigo que garda o código do estado civil desa persoa. Táboa de interpretación de códigos de estado civil:

Código	Descrición
S	Solteiro/a
C	Casado/a
D	Divorciado/a
V	Viúvo/a
U	Unión Libre/a
P	Separado/a

Solución

```
case vCodigo
  when 'S' then select 'Solteiro/a';
  when 'C' then select 'Casado/a';
  when 'D' then select 'Divorciado/a';
  when 'V' then select 'Viúvo/a';
  when 'U' then select 'Unión Libre';
  when 'P' then select 'Separado/a';
  else select 'Erro no código';
end case;
```

Para probar o funcionamento da sentenza *case*, pódese crear un procedemento almacenado cun único bloque de programación:

```
delimiter //
create procedure estado_civil ()
begin
  declare vCodigo varchar(1) default 'b';
  case vCodigo
    when 'S' then select 'Solteiro/a';
    when 'C' then select 'Casado/a';
    when 'D' then select 'Divorciado/a';
    when 'V' then select 'Viúvo/a';
    when 'U' then select 'Unión Libre';
    when 'P' then select 'Separado/a';
    else select 'Erro no código';
  end case;
end
//
delimiter ;
call estado_civil();
```

Resultado da execución:

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
Erro no código			
▶ Erro no código			

Tarefa 5. Utilizar a sentenza while

A tarefa consiste en escribir e comprobar que funcione un bloque de programación, que utiliza unha sentenza *while*, para calcular a cantidade de números pares que hai entre dous va-

lores que se gardan en dúas variables de tipo enteiro sen signo chamadas *vNumeroInicio* e *vNumeroFinal*.

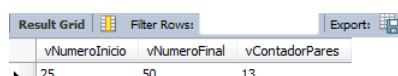
```
declare vNumero, vNumeroInicio, vNumeroFinal smallint unsigned;
declare vContadorPares smallint unsigned default 0;
set vNumeroInicio = 25;
set vNumeroFinal = 50;
set vNumero = vNumeroInicio;
while vNumero <= vNumeroFinal do
    if vNumero % 2 = 0 then
        set vContadorPares = vContadorPares + 1;
    end if;
    set vNumero = vNumero + 1;
end while;
```

Para probar o funcionamento da sentenza while, pódese crear un procedemento almacenado cun único bloque de programación:

/*Escribir un bloque de programación, utilizando unha sentenza while, que calcule a cantidade de números pares que hai entre dous valores que se gardan en dúas variables de tipo enteiro sen signo chamadas vNumeroInicio e vNumeroFinal*/

```
drop procedure if exists contarPares;
delimiter //
create procedure contarPares()
begin
    declare vNumero, vNumeroInicio, vNumeroFinal smallint unsigned;
    declare vContadorPares smallint unsigned default 0;
    set vNumeroInicio = 25;
    set vNumeroFinal = 50;
    set vNumero = vNumeroInicio;
    while vNumero <= vNumeroFinal do
        if vNumero % 2 = 0 then
            set vContadorPares = vContadorPares + 1;
        end if;
        set vNumero = vNumero + 1;
    end while;
    select vNumeroInicio, vNumeroFinal, vContadorPares;
end;
//
delimiter ;
call contarPares();
```

Resultado da execución:



vNumeroInicio	vNumeroFinal	vContadorPares
25	50	13

Tarefa 6. Utilizar a sentenza repeat

A tarefa consiste en escribir e comprobar que funcione un bloque de programación, que utiliza a sentenza *repeat*, para calcular o número de intentos realizados ata acertar o valor entre 1 e 100 almacenado nunha variable, xerando un número aleatorio en cada volta do bucle.

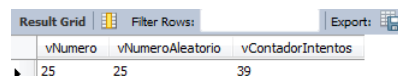
Solución

```
declare vNumero, vNumeroAleatorio tinyint unsigned default 0;
declare vContadorIntentos smallint unsigned default 0;
set vNumero = 25;
repeat
    /*seleccionar un número aleatorio entre 1 e 100*/
    set vNumeroAleatorio = floor(1 + (rand() * 99));
    set vContadorIntentos = vContadorIntentos + 1;
until vNumero = vNumeroAleatorio
end repeat;
```

Para probar o funcionamento da sentenza *repeat*, pódese crear un procedemento almacenado cun único bloque de programación:

```
/*Escribir un bloque de programación, utilizando a sentenza repeat, que calcule
o número de intentos realizados ata acertar o valor entre 1 e 100, almacenado
nunha variable, xerando un número aleatorio en cada execución do bucle.
*/
drop procedure if exists buscarNumero;
delimiter //
create procedure buscarNumero()
begin
    declare vNumero, vNumeroAleatorio tinyint unsigned default 0;
    declare vContadorIntentos smallint unsigned default 0;
    set vNumero = 25;
    repeat
        /*seleccionar un número aleatorio entre 1 e 100*/
        set vNumeroAleatorio = floor(1 + (rand() * 99));
        set vContadorIntentos = vContadorIntentos + 1;
    until vNumero = vNumeroAleatorio
    end repeat;
    select vNumero, vNumeroAleatorio, vContadorIntentos;
end;
//
delimiter ;
call buscarNumero();
```

Resultado da execución que dependerá dos números aleatorios xerados:



vNumero	vNumeroAleatorio	vContadorIntentos
25	25	39

Tarefa 7. Utilizar sentenzas preparadas

A tarefa consiste en crear dúas sentenzas preparadas:

- Tarefa 7.1. Crear unha sentenza preparada que insira unha columna na táboa *provincia* da base de datos *utilidades*, pasándolle os valores contidos nas variables de usuario chamadas *codigo* e *nome*.
- Tarefa 7.2. Crear unha sentenza preparada que mostre os nomes das táboas dunha base de datos. O nome da base de datos se obtén dunha variable chamada *nomeBD*.

Solución

■ Tarefa 7.1

```
/* Crear unha sentenza preparada que insira unha columna na táboa departamento
da base de datos practicas1, pasándolle os valores para as columnas en variables
```

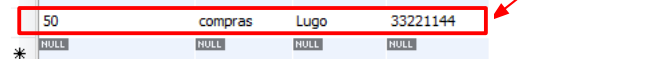
```

de usuario.*/
prepare insertDepartamento from 'insert into practicas1.departamento values (?, ?, ?, ?)';
set @codigo = '50';
set @nome = 'compras';
set @localizacion = 'Lugo';
set @nssXefe = '33221144';
execute insertDepartamento using @codigo, @nome, @localizacion, @nssXefe;
select * from practicas1.departamento;
deallocate prepare insertDepartamento;

```

Resultado da execución:

codigoDepartamento	nome	localizacion	nssXefe
10	xerencia	Lugo	33221144
20	producción1	Lugo	33221144
30	producción 2	Ourense	3202588520
40	producción3	Coruña	150251236
50	compras	Lugo	33221144

* 

■ Tarefa 7.2

```

/* Crear unha sentenza preparada que mostre os nomes das táboas dunha base
de datos. O nome da base de datos se obtén dunha variable chamada nomeBD.*/
set @nomeBD = 'practicas1';
set @senteza = concat('select table_name from information_schema.tables where ta-
ble_schema = ', @nomeBD, '');
prepare tablasBD from @senteza;
execute tablasBD;
deallocate prepare tablasBD;

```

Resultado da execución:

table_name
actuacion
ciclista
concerto
contador
departamento
empregado
equipo
fabricante
grupo
nacionalidade
película
piso
representante
xenaro