Dirección Xeral de Educación, Formación Profesional e Innovación Educativa

Material para

a formación profesional inicial

A04. Cursores

|  |  |  |
| --- | --- | --- |
| Familia profesional | IFC | Informática e comunicacións |
| Ciclo formativo | CSIFC03  CSIFC02 | Desenvolvemento de aplicacións web  Desenvolvemento de aplicacións multiplataforma |
| Grao |  | Superior |
| Módulo profesional | MP0484 | Bases de datos |
| Unidade didáctica | UD07 | Programación de bases de datos |
| **Actividade** | **A04** | **Cursores** |
| Autores |  | Marta Fernández García  María del Carmen Fernández Lameiro  Miguel Fraga Vila  María Carmen Pato González  Andrés del Río Rodríguez |
| Nome do arquivo |  | CSIFC02\_ MP0484\_V000704\_U07\_A04\_Cursor.docx |
| © 2015 Xunta de Galicia.  Consellería de Cultura, Educación e Ordenación Universitaria.  Este traballo foi realizado durante unha licenza de formación retribuída pola Consellería de Cultura, Educación e Ordenación Universitaria e ten licenza CreativeCommons BY-NC-SA (recoñecemento - non comercial - compartir igual). Para ver unha copia desta licenza, visitar a ligazón http://creativecommons.org/licenses/by-nc-sa/3.0/es/. | | |

Índice

[1. Ficha técnica 3](#_Toc439856594)

[Contexto da actividade 3](#_Toc439856595)

[Título da actividade 4](#_Toc439856596)

[Resultados de aprendizaxe do currículo 4](#_Toc439856597)

[Obxectivos didácticos e título e descrición da actividade 4](#_Toc439856598)

[Criterios de avaliación 4](#_Toc439856599)

[Contidos 4](#_Toc439856600)

[Actividades de ensino e aprendizaxe e de avaliación, métodos, recursos e instrumentos de avaliación 5](#_Toc439856601)

[2. A04. Cursores 6](#_Toc439856602)

[2.1 Introdución 6](#_Toc439856603)

[2.1.1 Obxectivos 6](#_Toc439856604)

[2.1.2 Software 6](#_Toc439856605)

[2.1.3 Bases de datos de traballo 7](#_Toc439856606)

[2.2 Actividade 7](#_Toc439856607)

[2.2.1 Cursores 7](#_Toc439856608)

[2.2.1.1 Sentenza DECLARE ... CURSOR 7](#_Toc439856609)

[2.2.1.2 Sentenza OPEN 7](#_Toc439856610)

[2.2.1.3 Sentenza FETCH 8](#_Toc439856611)

[2.2.1.4 Sentenza CLOSE 8](#_Toc439856612)

[2.2.2 Utilización de cursores 9](#_Toc439856613)

[2.2.3 Cursores e sentenzas preparadas 11](#_Toc439856614)

[2.3 Tarefas 12](#_Toc439856615)

[2.3.1 Tarefa 1. Identificar os pasos e sentenzas para utilizar cursores 12](#_Toc439856616)

[Solución 13](#_Toc439856617)

[2.3.2 Tarefa 2. Utilizar cursores en programas almacenados 13](#_Toc439856618)

[Solución 13](#_Toc439856619)

[3. Materiais 15](#_Toc439856620)

[3.1 Documentos de apoio ou referencia 15](#_Toc439856621)

[3.2 Recursos didácticos 15](#_Toc439856622)

[3.3 Material auxiliar 15](#_Toc439856623)

[4. Avaliación 16](#_Toc439856624)

[Criterios de avaliación 16](#_Toc439856625)

[Modelo de probas para TO.8 16](#_Toc439856626)

[Proba para CA7.11 17](#_Toc439856627)

[Exemplo de solución para entregar en papel 17](#_Toc439856628)

[Exemplo de lista de valoración para TO.8 19](#_Toc439856629)

1. Ficha técnica

Contexto da actividade

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Módulo | Duración  horas | Unidade didáctica. | Sesións 50´ | Actividades | Sesións 50´ |
| MP0484. Bases de datos | 187 | UD01. Bases de datos e sistemas de almacenamento da información | 12 | A01. Bases de datos e sistemas de almacenamento da información | 7 |
| A02. Introdución aos sistemas xestores de bases de datos | 5 |
| UD02. Deseño conceptual de bases de datos | 38 | A01. Fases do deseño de bases de datos | 3 |
| A02.Descrición e representación gráfica dos elementos do Modelo Entidade-Interrelación (MER) | 10 |
| A03. Descrición e representación gráfica dos elementos do Modelo Entidade-Interrelación Estendido(MERE) | 9 |
| A04. Construción e validación do modelo de datos | 11 |
| A05 Notación Martin e ferramentas CASE | 5 |
| UD03. Deseño lóxico de base de datos | 32 | A01. O deseño lóxico empregando o modelo relacional | 15 |
| A02. Normalización de relacións | 10 |
| A03. Operacións do MR: álxebra e cálculo | 7 |
| UD04. Deseño físico de bases de datos | 28 | A01. Creación da estrutura de bases de datos relacionais | 13 |
| A02. Modificación da estrutura de bases de datos relacionais | 9 |
| A03. Verificación e probas | 3 |
| A04. Utilización de ferramentas gráficas para a creación da estrutura de bases de datos relacionais | 3 |
| UD05. Consultas de datos | 39 | A01. Consultas simples cunha táboa. | 12 |
| A02. Consultas con datos de máis dunha táboa | 10 |
| A03. Consultas resumo con agrupamentos | 7 |
| A04. Consultas con subconsultas | 7 |
| A05. Optimización de consultas | 3 |
| UD06. Tratamento de datos | 24 | A01. Manipulación de datos | 12 |
| A02. Vistas | 5 |
| A03. Transaccións e bloqueos | 7 |
| UD07. Programación de bases de datos | **22** | A01. Introdución á programación con SQL | 4 |
| A02. Rutinas almacenadas | 9 |
| A03. Disparadores e eventos | 5 |
| **A04. Cursores** | **4** |
| UD08. Administración de bases de datos | 19 | A01. Copias de seguridade e intercambio de datos entre SXBDs | 10 |
| A02. Índices e xestión de usuarios | 9 |
| UD09. Bases de datos obxecto-relacionais | 10 | A01. Uso de bases de datos obxecto-relacionais | 10 |

NOTA: Esta actividade está vinculada á programación recollida no arquivo CSIFC02\_MP0487\_V000700\_UD07\_ProgramacionBD.pdf

Título da actividade

|  |  |  |  |
| --- | --- | --- | --- |
| Nº | Título | Descrición | Duración |
| A04 | Cursores. | Utilizaranse cursores para manexar a información obtida nas consultas que devolven máis dunha fila. | 4 |

Resultados de aprendizaxe do currículo

|  |  |
| --- | --- |
| Resultados de aprendizaxe do currículo | Completo |
| * RA7 -  Desenvolve procedementos almacenados e guións de sentenzas, para o que utiliza e avalía as sentenzas da linguaxe incorporada no sistema xestor de bases de datos. | Si |

Obxectivos didácticos e título e descrición da actividade

| Obxectivos específicos | | Actividade | | Descrición básica | Duración |
| --- | --- | --- | --- | --- | --- |
| O4.1 | Utilizar cursores. | A04 | Cursores | Utilizaranse cursores para manexar a información obtida nas consultas que devolven máis dunha fila. | 4 |

Criterios de avaliación

|  |
| --- |
| Criterios de avaliación |
| * CA7.11 Utilizáronse cursores. |

Contidos

| Contidos |
| --- |
| * Cursores. |

Actividades de ensino e aprendizaxe e de avaliación, métodos, recursos e instrumentos de avaliación

| Que e para que | Como | | | Con que | Como e con que se valora | Duración (sesións) |
| --- | --- | --- | --- | --- | --- | --- |
| Actividade (título e descrición) | Profesorado (en termos de tarefas) | Alumnado (tarefas) | Resultados  ou produtos | Recursos | Instrumentos e procedementos de avaliación |
| A04. Cursores   * Utilizaranse cursores para manexar a información obtida nas consultas que devolven máis dunha fila. | * **Tp4.1** Explicación da sintaxe de sentenzas SQL relacionadas coa creación e utilización de cursores. | * **Ta4.1** Creación e utilización de cursores. | * Guións de sentenzas SQL coa creación e utilización de cursores. * Resultado da utilización dos cursores. | * Material didáctico subministrado polo profesorado en papel e/ou formato dixital. * Ordenadores con conexión a Internet, que terán instalado MySQL e MySQL Workbench. * Proxector. * Manual de referencia de MySQL. |  | 3.5 |
|  | * **Ta4.2** Tarefa de avaliación utilizando o instrumento TO.8. | * Exame en papel e en formato dixital. | * Máquina virtual para exame que terá instalado MySQL e MySQL Workbench. * Manual de referencia de MySQL. | **TO.8** Documento de rexistro de creación e utilización de cursores. Escala de valores (observación indirecta). | 0.5 |

1. A04. Cursores
   1. Introdución
      1. Obxectivos

O obxectivo desta actividade é:

* Utilizar cursores en programas almacenados.
  + 1. Software

Utilizarase a plataforma WAMP (Windows-Apache-MySQL-PHP) WampServer 2.5 (última versión estable en outubro 2015), que inclúe MySQL Community Edition 5.6.17 como SXBDR (Sistema Xestor de Bases de Datos Relacional). As razóns de utilización deste software son que:

* É software libre, polo que o alumnado poderá descargalo de forma gratuíta e utilizalo legalmente na súa casa.
* É unha forma sinxela de facer a instalación do software necesario para desenvolver aplicacións web.
* Páxina oficial de  WampServer: <http://www.wampserver.com>
* Páxina oficial de MySQL: <https://www.mysql.com/>

Utilizarase MySQL Workbench 6.3 como ferramenta cliente gráfica xa que é a recomendada por MySQL en outubro de 2015, aínda que tamén poderían utilizarse outras como phpMyAdmin, EMS MyManager, ou MySQL Query Browser.

O resultado das probas dos guións SQL desta actividade, ilustrarase normalmente cunha imaxe capturada da zona de manipulación de datos de Workbench. Para completar esa información, por exemplo, cando non é posible mostrar o resultado completo, mostraranse outras zonas de Workbench, como por exemplo a zona de saída (output) con información do estado da execución do guión e o número de filas que devolve.

* En <https://www.mysql.com/products/workbench/> pode obterse información detallada sobre a ferramenta MySQL Workbench e descargar o software.
* En <http://dev.mysql.com/doc/index-gui.html> pode descargarse o manual de MySQL Workbench.
* O material anexo a esta actividade inclúe unha Guía básica de MySQL Workbench 6.3.
  + 1. Bases de datos de traballo

As bases de datos *traballadores* e *tendaBD*, utilizaranse nalgúns exemplos e tarefas desta actividade. Antes de empezar a probar os exemplos ou realizar as tarefas, hai que executar os scripts de creación no servidor e poñer en uso as bases de datos cando corresponda. Os scritps atópanse no cartafol anexo a esta actividade descrito no apartado '3.3 Material auxiliar'.

* 1. Actividade
     1. Cursores

Os cursores serven para procesar fila a fila o resultado dunha consulta SELECT, dentro dos programas almacenados. Cando a sentenza SELECT devolve unha única fila, pódense asignar os valores das columnas a variables utilizando a sentenza SELECT ... INTO, e non é necesario o uso de cursores.

Estas estruturas só deben utilizarse cando non se teña outra forma de manipular conxuntos de datos cunha soa sentenza.

Cada SXBDR ten a súa forma de declarar un cursor e incluso os seus propios tipos de cursores (Oracle PL/SQL, Transact, etc).

MySQL permite utilizar cursores dende a versión 5.0. Os cursores en MySQL son de só lectura, é dicir, pódese ler a información que conteñen, pero non se pode escribir neles.

Outra característica dos cursores en MySQL é que a información que conteñen só pode ser procesada de maneira secuencial fila a fila, empezando pola primeira fila, e non é posible acceder a unha fila directamente sen pasar polas anteriores. A cada fila pódeselle aplicar un bloque de sentenzas SQL.Para utilizar cursores en MySQL hai que seguir catro pasos:

* Declaración.
* Apertura.
* Lectura.
* Peche.
  + - 1. Sentenza DECLARE ... CURSOR

A sentenza DECLARE ... CURSOR permite declarar un cursor, asignarlle un nome, e relacionalo cunha sentenza SELECT. Sintaxe:

DECLARE nome\_cursor CURSOR FOR sentenza\_SELECT

O cursor debe declarase (definirse) antes de usarse. No proceso de declaración non se recuperan datos, simplemente asóciase a unha sentenza SELECT. A declaración dos cursores ten que ir sempre despois da declaración de variables e de condicións, pero antes da declaración de manipuladores. Nun programa almacenado pódense declarar máis dun cursor.

* + - 1. Sentenza OPEN

A sentenza OPEN permite abrir un cursor para poder procesar a información resultante da consulta SELECT. Sintaxe:

OPEN nome\_cursor

A operación de apertura ten que facerse despois de declarar o cursor e antes de facer ningunha operación con el. Esta operación recupera os datos que resultan de executar a sentenza SELECT que ten asociada.

Cando se abre o cursor, créase un punteiro que sinala á primeira fila de resultados. Este punteiro é utilizado polo sistema para sinalar a fila que se vai a ler cando se execute unha operación de lectura.

* + - 1. Sentenza FETCH

A sentenza FETCH permite ler as filas obtidas pola consulta SELECT asociada a un cursor. Cada vez que se executa esta sentenza, móvense os valores das columnas correspondente a unha fila do conxunto de resultados, a un conxunto de variables e avanza o punteiro á fila seguinte. Sintaxe:

FETCH nome\_cursor INTO nome\_variable [, nome\_variable ] …

Despois da palabra INTO teñen que especificarse tantas variables como columnas devolve a sentenza SELECT e teñen que estar definidas previamente co mesmo tipo de dato que a columna da que vai recibir o valor.

A lectura de resultados é sempre secuencial. A primeira vez que se executa unha sentenza FETCH, despois de abrir o cursor, accédese á primeira fila obtida pola consulta, a segunda vez que se executa accédese á segunda fila, e así sucesivamente.

Para poder recorrer todas as filas que devolve a sentenza SELECT é necesario executar un número variable de veces a sentenza FETCH, en función do número de filas que se devolvan. Para facer isto, o lóxico é utilizar unha estrutura repetitiva (WHILE ou REPEAT) e incluír a sentenza FETCH dentro do bucle, que debe finalizar cando xa non queden filas do resultado por ler.

Cando xa non quedan filas por ler no cursor, e se executa unha sentenza FETCH, prodúcese un erro asociado ao código SQLSTATE '02000', ou ben á condición de erro 'NOT FOUND', polo que para establecer a condición para que finalice a execución do bucle hai que definir un manipulador asociado a ese código de erro.

Exemplo de manipulador para controlar o final do bucle:

**declare** **continue** handler **for** **SQLSTATE** '02000' **set** vFinal **=** 1**;**

Esta sentenza de declaración de manipulador indica que se ocorre o erro con código SQLSTATE '02000' non finalice a execución do programa almacenado (por ser tipo 'continue'), e que se lle asigne o valor 1 á variable *vFinal*, que ten que estar previamente declarada. Esta variable pode ser utilizada para establecer a condición para finalizar o bucle, é dicir, o bucle debe finalizar cando a variable tome o valor 1.

Outra forma de declarar o manipulador que produce o mesmo resultado cá anterior é:

**declare** **continue** handler **for** **NOT** **FOUND** **set** vFinal **=** 1**;**

* + - 1. Sentenza CLOSE

A sentenza CLOSE permite pechar un cursor e limpar os espazos de memoria que utiliza.

CLOSE nome\_cursor

O cursor péchase automaticamente no momento que finaliza o programa almacenado que o contén, aínda que non se execute a sentenza CLOSE. Mentres o programa non finalice, un cursor pode ser pechado e aberto as veces que se necesite.

* Tarefa 1. Identificar os pasos e sentenzas para utilizar cursores.
  + 1. Utilización de cursores

O proceso de utilización de cursores require facer unha serie de operacións que son obrigatorias sempre, e polo tanto pódese facer un esqueleto coas sentenzas SQL necesarias para utilizar cursores.

Cando se utiliza MySQL Workbench para editar os procedementos almacenados que utilizan cursores, pódese crear un fragmento de código (*snippet*) e gardalo para utilizalo cada vez que se escribe o código dun cursor. Para máis información sobre snippets, pódese consultar a Guía básica de MySQL Workbench 6.3 anexionada a esta actividade.

Exemplo de contido do snippet esqueleto\_cursor:

**begin**

-- Declaración de variables

**declare** vFinal **integer** **default** 0**;**

/\* aquí decláranse tamén as variables para almacenar os datos contidos

nas columnas da consulta\*/

-- Declaración do cursor

**declare** nomeCursor **cursor** **for**

sentenza\_SELECT**;**

-- Declaración do manipulador para controlar o final da lectura do cursor

**declare** **continue** handler **for** **sqlstate** '02000' **set** vFinal**=**1**;**

-- Abrir cursor: recupéranse os datos da consulta

**open** nomeCursor**;**

-- Bucle para procesar os datos asociados ao cursor, ata que non queden filas

/\*Sábese que non quedan filas coando se produce o erro código sqlstate '02000

nese momento asígnaselle á variable vFinal o valor 1\*/

**repeat**

-- Lectura dos datos dunha fila do cursor e asignación a variables

**fetch** nomeCursor **into** lista\_variables**;**

**if** vFinal **=** 0 **then**

-- Aquí van as sentenzas para procesar unha fila do cursor

**end** **if;**

-- Condición de final da execución do bucle: cando vFinal tome o valor 1

until vFinal **=** 1 **end** **repeat;**

-- Peche do cursor, para liberar memoria

**close** nomeCursor**;**

**end**

Exemplo de utilización de cursores:

/\*

u704exemplo\_demoCursor1.sql

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

NOME RUTINA: traballadores.sp\_demoCursor1 (procedemento)

DATA CREACIÓN: 23/11/2015

AUTOR: Grupo licenza 2015

TAREFA A AUTOMATIZAR: - Utilizando un cursor, que conta os empregados agrupados por

departamento, actualizar a columna depEmpregados (número de

empregados do departamento) na táboa departamento no caso que

o número de empregados contados non coincida co contido actual

da columna depEmpregados.

- Contar o número de departamentos que foron actualizados.

PARAMETROS REQUERIDOS: - Non require parámetros.

RESULTADOS PRODUCIDOS: - Mostrar na pantalla a data na que se fai a actualización, e

o número de departamentos que foron actualizados.

- Columna depEmpregados actualizada na táboa departamento.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*/

**drop** **procedure** traballadores**.**sp\_demoCursor1**;**

**delimiter** **//**

**create** **procedure** traballadores**.**sp\_demoCursor1**()**

**begin**

-- Declaración de variables:

-- vDepartamento e vContador almacenan os valores das columnas da consulta

**declare** vDepartamento**,** vContador **integer;**

-- vFinal utilízase para controlar o momento que non hai máis filas

-- vFModificados é un contador dos departamentos actualizados

**declare** vFinal**,** vModificados **integer** **default** 0**;**

-- Declaración de cursores

**declare** cursorEmpregados **cursor** **for**

**select** empDepartamento**,count(\*)**

**from** traballadores**.**empregado

**group** **by** empDepartamento**;**

-- Declaración do manipulador para controlar o final da lectura do cursor

**declare** **continue** handler **for** **sqlstate** '02000' **set** vFinal**=**1**;**

-- Abrir cursor: recupéranse os datos da consulta

**open** cursorEmpregados**;**

-- Bucle para procesar os datos asociados ao cursor, ata que non queden filas

**repeat**

-- Lectura dos datos dunha fila do cursor e asignación a variables

**fetch** cursorEmpregados **into** vDepartamento**,** vContador**;**

**if** vFinal **=** 0 **then**

-- Procesamento dunha fila do cursor

**if** vContador **<>** **(select** depEmpregados

**from** traballadores**.**departamento

**where** depNumero**=**vDepartamento**)**

**then**

-- contar departamento modificado

**set** vModificados **=** vModificados **+** 1**;**

-- modificar os datos do departamento

**update** traballadores**.**departamento **set** depEmpregados**=**vContador

**where** depNumero**=**vDepartamento**;**

**end** **if;**

**end** **if;**

-- Condición de final da execución do bucle: cando vFinal tome o valor 1

until vFinal **=** 1 **end** **repeat;**

-- Peche do cursor, para liberar memoria

**close** cursorEmpregados**;**

-- Mostra número de departamentos modificados

**select** now**()** **as** 'Data actualización'**,** vModificados **as** 'Departamentos actualizados'**;**

**end**

**//**

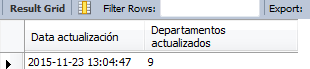
**delimiter** **;**

Este procedemento almacenado utiliza o cursor *cursorEmpregados* asociado a unha consulta sobre a táboa *empregado* que devolve o número do departamento e o contador de empregados que traballan nese departamento. Cando se le unha fila do cursor, compárase o valor da columna *depEmpregados* da táboa *departamento* co valor que devolve o contador da consulta asociada ao cursor. Se non coinciden os valores, actualízase o contido da columna *depEmpregados* co resultado do contador, e contabilízase a actualización feita empregando unha variable tipo contador. Ao finalizar o proceso, móstrase a data e hora do sistema e o número de departamentos que foron actualizados.

Pódense executar as seguintes sentenzas para facer unha proba do funcionamento do procedemento e ver o resultado na zona *Result Grid* en Workbench.

**update** traballadores**.**departamento **set** depEmpregados**=**0**;**

**call** sp\_demoCursor1**();**



* Tarefa2. Utilizar cursores en programas almacenados.
  + 1. Cursores e sentenzas preparadas

A combinación do uso de cursores e de sentenzas preparadas proporciona aos administradores de bases de datos unha ferramenta moi potente para automatizar tarefas propias da administración.

Exemplo de cursor combinado con sentenzas preparadas:

/\*

u704exemplo\_demoCursor2.sql

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

NOME RUTINA: utilidades.analizartaboas (procedemento)

DATA CREACIÓN: 23/11/2015

AUTOR: Grupo licenza 2015

TAREFA A AUTOMATIZAR: - Analizar todas as táboas dunha base de datos que se pasa

como parámetro, utilizando a sentenza:

'analyze table nome\_táboa'

PARAMETROS REQUERIDOS: - IN: nome da base de datos

RESULTADOS PRODUCIDOS: - Executar a sentenza 'analyze table nome\_táboa' para cada

táboa.

- Saída por pantalla do resultado da análise.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*/

**drop** **procedure** **if** **exists** utilidades**.**analizartaboas**;**

**delimiter** **//**

**create** **procedure** utilidades**.**analizartaboas**(**pBaseDatos **varchar(**64**))**

**begin**

-- Declaración de variables:

-- vTaboa almacena os nomes da táboas que se obteñen na consulta

**declare** vTaboa **varchar(**64**)** **character** **set** utf8**;**

-- vFinal utilízase para controlar o momento que non hai máis filas

**declare** vFinal **integer** **default** 0**;**

-- Declaración de cursores

**declare** cursorTaboas **cursor** **for**

**select** table\_name

**from** information\_schema**.**tables

**where** table\_schema**=**pBaseDatos**;**

-- Declaración de manipuladores:

-- Declaración do manipulador para controlar o final da lectura do cursor

**declare** **continue** handler **for** **sqlstate** '02000' **set** vFinal**=**1**;**

-- Abrir cursor: recupéranse os datos da consulta

**open** cursorTaboas**;**

-- Comprobar se existe a base de datos

**if** **(select** **count(\*)**

**from** information\_schema**.**schemata

**where** schema\_name **=** pBaseDatos**)** **=** 0

**then**

**select** 'A base de datos non existe'**;**

**else**

-- No caso de que exista a base de datos:

-- Bucle para procesar os datos asociados ao cursor, ata que non queden filas

**repeat**

-- Lectura dos datos dunha fila do cursor e asignación a variables

**fetch** cursorTaboas **into** vTaboa**;**

**if** vFinal **=** 0 **then**

-- Procesamento dunha fila do cursor: analizar a táboa

**set** @sentenza **=** **concat(**'analyze table '**,**pBaseDatos**,**'.'**,**vTaboa**);**

**prepare** analizar **from** @sentenza**;**

**execute** analizar**;**

**end** **if;**

-- Condición de final da execución do bucle: cando vFinal tome o valor 1

until vFinal **=** 1 **end** **repeat;**

-- borrado da sentenza preparada, para liberar memoria

**drop** **prepare** analizar**;**

-- Peche do cursor, para liberar memoria

**close** cursorTaboas**;**

**end** **if;**

**end**

**//**

**delimiter** **;**

Este tipo de procedementos almacenados utilizan a información do dicionario de datos para poder automatizar operacións sobre obxectos das bases de datos. No caso do exemplo execútase a sentenza ANALYZE TABLE para facer un mantemento dos índices e mellorar o rendemento das consultas. A execución desta sentenza produce o bloqueo da táboa mentres se fai o análise, e mostra como saída unha mensaxe informando do estado da operación. Como a sintaxe da sentenza non admite utilizar variables para poñer o nome da táboa, hai que recorrer ás sentenzas preparadas.

Exemplo de execución do procedemento almacenado:

**call** utilidades**.**analizartaboas**(**'traballadores'**);**



Pódese automatizar a execución do procedemento almacenado, para que se execute cada certo tempo, creando un evento que fixe o calendario de execución.

Exemplo:

-- habilitar o planificador de eventos

**set** **global** event\_scheduler **=** **on;**

-- crear o evento

**delimiter** **//**

**create** **event** traballadores**.**analizarTaboas

**on** **schedule** **every** 1 **month**

**starts** now**()** **+** **interval** 15 **day**

**do**

**begin**

**call** utilidades**.**analizartaboas**(**'traballadores'**);**

**end**

**//**

**delimiter** **;**

* 1. Tarefas

As tarefas propostas son as seguintes:

* Tarefa 1. Identificar os pasos para utilizar cursores.
* Tarefa 2. Utilizar cursores en programas almacenados.
  + 1. Tarefa 1. Identificar os pasos e sentenzas para utilizar cursores

A tarefa consiste en nomear e explicar brevemente os pasos e sentenzas necesarios para poder utilizar cursores.

Solución

* Declarar: Asignar un nome ao cursor e asociarlle unha sentenza SELECT. Faise coa sentenza CREATE....CURSOR.
* Abrir: Recuperar datos da consulta. Faise coa sentenza OPEN.
* Ler: Ler fila a fila o resultado da consulta. Faise coa sentenza FETCH.
* Pechar: Pechar o acceso ao cursor e liberar memoria. Pode facerse coa sentenza CLOSE.
  + 1. Tarefa 2. Utilizar cursores en programas almacenados

A tarefa consiste en crear un procedemento almacenado na base de datos *tendaBD*, ao que se lle pasen como parámetros de entrada dúas datas, e mostre o *id* do empregado, e a suma dos importes das ventas do empregado que fixo vendas por un maior importe entre esas datas.

Solución

/\*

u7a4tarefa2.sql

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

NOME RUTINA: tendabd.vendasEmpregado (procedemento)

DATA CREACIÓN: 23/11/2015

AUTOR: Grupo licenza 2015

TAREFA A AUTOMATIZAR: - Utilizar un cursor, que selecciona o id dos empregados e

a suma dos importes de todas as vendas feitas por eles

entre dúas datas que se pasan como parámetros de entrada,

tendo en conta a información contida na táboa detalle\_vendas.

- Compara a suma dos importe de todos os empregados, e

mostrar o id e o importe total das ventas do empregado que

fixo vendas por un importe maior entre esas datas.

PARAMETROS REQUERIDOS: - data\_inicio e data\_final.

RESULTADOS PRODUCIDOS: - Mostrar en pantalla o id e o importe total das ventas do

empregado que fixo vendas por un importe maior entre esas

datas.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*/

delimiter **//**

**create** **procedure** tendabd**.**vendasEmpregado**(**data\_inicio **date,** data\_final **date)**

**begin**

-- Declaración de variables:

-- vEmpId e vSumaVendas almacenan os valores das columnas da consulta

-- vEmpIdMax e vSumaMax almacenan os datos do empregado co maior importe de vendas

**declare** vEmpId**,** vEmpIdMax **smallint;**

**declare** vSumaVendas**,** vSumaMax **decimal(**11**,**2**)** **default** 0**;**

-- vFinal utilízase para controlar o momento que non hai máis filas

**declare** vFinal**,** vModificados **integer** **default** 0**;**

-- Declaración de cursores

**declare** cursorVendasEmpregado **cursor** **for**

**select** ven\_empregado**,**

**sum(**dev\_cantidade**\***dev\_prezo\_unitario**-**

**(**dev\_cantidade**\***dev\_prezo\_unitario**)\***dev\_desconto**/**100**)**

**from** tendabd**.**vendas **join** tendabd**.**detalle\_vendas **on** ven\_id **=** dev\_venda

**where** ven\_data **between** data\_inicio **and** data\_final

**group** **by** ven\_empregado**;**

-- Declaración de manipuladores:

-- Declaración do manipulador para controlar o final da lectura do cursor

**declare** **continue** handler **for** **not** **found** **set** vFinal**=**1**;**

-- Abrir cursor: recupéranse os datos da consulta

**open** cursorVendasEmpregado**;**

-- Bucle para procesar os datos asociados ao cursor, ata que non queden filas

**repeat**

-- Lectura dos datos dunha fila do cursor e asignación a variables: gárdase

-- o id do empregado e a suma dos importes das ventas que fixo entre as datas

**fetch** cursorVendasEmpregado **into** vEmpId**,** vSumaVendas**;**

**if** vFinal **=** 0 **then**

-- se o empregado actual ten mellores resultados que o mellor ata o momento

-- gárdanse os seus datos nas variables vEmpIdMax e vSumaMax

**if** vSumaVendas**>=**vSumaMax **then**

**set** vSumaMax **=** vSumaVendas**;**

**set** vEmpIdMax **=** vEmpId**;**

**end** **if;**

**end** **if;**

-- Condición de final da execución do bucle: cando vFinal tome o valor 1

until vFinal **=** 1 **end** **repeat;**

-- Peche do cursor, para liberar memoria

**close** cursorVendasEmpregado**;**

-- Mostra o código e total acumulado das vendas do mellor empregado

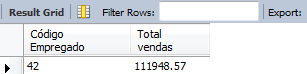
**select** vEmpIdMax **as** 'Código Empregado'**,** vSumaMax **as** 'Total vendas'**;**

**end//**

delimiter **;**

Exemplo de execución do procedemento almacenado:

**call** tendabd**.**vendasEmpregado**(**'2015-05-01'**,** '2015-05-31'**);**



Este procedemento está feito con fins educativos, para ver o funcionamento dun cursor. Pódese obter a información solicitada escribindo unha consulta complexa.

1. Materiais
   1. Documentos de apoio ou referencia

* HUESO IBAÑEZ, Luis. *Bases de datos*. Ciclos Formativos Ra-Ma, 2012.
* SANCHEZ, Jorge. *Manual de Gestión e bases de datos*.

<http://www.jorgesanchez.net/bd/sgbd.html>

* *Bases de datos. Máster oficial de Software libre*. UOC.

<http://ocw.uoc.edu/informatica-tecnologia-y-multimedia/bases-de-datos/Course_listing>

* *Manual de referencia de MySQL 5.6*. MySQL. <http://dev.mysql.com/doc/>
* GROFF, James.*SQL.* McGraw-Hill.
* *Cursores en MySQL.* Hermosa Programacion.

[http://www.hermosaprogramacion.com/2014/06/mysql-cursores](http://www.hermosaprogramacion.com/2014/06/mysql-cursores/)

* 1. Recursos didácticos
* Ordenadores con conexión a Internet, que terán instalado MySQL e MySQL Work-bench.
* Material didáctico subministrado polo profesorado en papel e/ou formato dixital.
* Máquina virtual para exame que terá instalado MySQL e MySQL Workbench.
* Manual de referencia de MySQL.
* Proxector.
  1. Material auxiliar

O material auxiliar anexo a esta actividade está almacenado na carpeta CSIFC02\_MP0484\_V000704\_UD07\_A04\_Cursores\_Anexo que contén:

* O arquivo GuiaWorkbench.docx que é unha guía básica de MySQL Workbench 6.3.
* O arquivo V000704\_scriptsSQL.zip comprimido cos scripts necesarios para crear as bases de datos empregadas nos exemplos, tarefas e proba de avaliación da actividade.

1. Avaliación

Criterios de avaliación

|  |  |  |  |
| --- | --- | --- | --- |
| Criterios de avaliación seleccionados  para esta actividade | Tipo de  evidencia de aprendizaxe | Instrumento de avaliación | Peso na cualificación  da UD |
| * CA7.11 Utilizáronse cursores. | * Exame en papel e en formato dixital. | * **TO.8** Documento de rexistro de creación e utilización de cursores. Escala de valores (observación indirecta). | 12 |

Modelo de probas para TO.8

Proponse unha proba cun apartado de tipo práctico. A solución deberá ser entregada polo alumnado nun documento escrito e en dixital.

* O documento escrito debe conter:
* Código para crear e utilizar o cursor.
* Imaxe capturada dende o cliente Workbench que permita comprobar o resultado do funcionamento do procedemento almacenado con cursor.
* O arquivo dixital debe conter:
* Arquivo .sql cos códigos solicitados.

A proba realizarase na base de datos *liga\_acb*. Subminístrase o arquivo *ligaAcb.sql* co script que permite crear esa base de datos. A estrutura da base de datos é a seguinte:

* Táboa *equipo*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nome columna | Tipo | Null | Clave | Observacións |
| codigo | char(4) |  |  | Código do equipo. Para enlazar con outras táboas |
| nome\_club | varchar(80) |  |  | Nome do club ao que pertence o equipo |
| pavillon | varchar(70) |  |  | Nome do pavillón no que xogan os partidos como local |
| capacidade\_pavillon | smallint unsigned |  |  | Número de espectadores que entran no seu pavillón |
| web | varchar(100) |  |  | Url da web |
| nome\_equipo | varchar(80) |  |  | Nome oficial do equipo na liga |

* Táboa *xogador*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nome columna | Tipo | Null | Clave | Observacións |
| codigo\_equipo | char(4) |  |  | Código do equipo no que xoga o xogador |
| numero | tinyint unsigned |  |  | Número de dorsal do xogador |
| nome | char(30) |  |  | Nome do xogador |
| posicion | char(1) |  |  | Posición na que xoga: B = base, P = pivot, ...... |
| nacionalidade | char(3) |  |  | Nacionalidade do xogador |
| ficha | char(3) |  |  | Tipo de ficha: EXT=estranxeiro EUR=europeo JFL=nacional |
| estatura | Decimal(3,2) |  |  | Estatura do xogador en metros |
| idade | tinyint unsigned |  |  | Idade do xogador |
| temporadas | tinyint |  |  | Número de tempadas que leva no equipo |

* Táboa *partido*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nome columna | Tipo | Null | Clave | Observacións |
| id | smallint unsigned | N | **P** | Un identificador para cada partido. AUTOINCREMENTAL |
| xornada | smallint unsigned |  |  | Número da xornada na que se xoga o partido |
| equipo\_local | char(4) |  |  | Código do equipo que xoga como local |
| equipo\_visitante | char(4) |  |  | Código do equipo que xoga como visitante |
| puntos\_local | tinyint unsigned |  |  | Puntos conseguidos polo equipo local no partido |
| puntos\_visitante | tinyint unsigned |  |  | Puntos conseguidos polo equipo visitante no partido |

* Táboa *estatistica*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nome columna | Tipo | Null | Clave | Observacións |
| equipo | char(4) |  |  | Código do equipo |
| xogados\_local | tinyint unsigned |  |  | Número de partidos xogados como local |
| xogados\_visitante | tinyint unsigned |  |  | Número de partidos xogados como visitante |
| ganados\_local | tinyint unsigned |  |  | Número de partidos ganados como local |
| ganados\_visitante | tinyint unsigned |  |  | Número de partidos ganados como visitante |
| puntos\_favor\_local | smallint |  |  | Número de puntos conseguidos xogando como local |
| puntos\_favor\_visitante | smallint |  |  | Número de puntos conseguidos xogando como visitante |
| puntos\_contra\_local | smallint |  |  | Número de puntos recibidos xogando como local |
| puntos\_contra\_visitante | smallint |  |  | Número de puntos recibidos xogando como visitante |

* Táboa *clasificacion*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nome columna | Tipo | Null | Clave | Observacións |
| posicion | tinyint unsigned |  |  | Nº de orde que ocupa o equipo na clasificación |
| equipo | char(4) |  |  | Código do equipo |
| xogados | tinyint unsigned |  |  | Número de partidos xogados |
| ganados | tinyint unsigned |  |  | Número de partidos ganados |
| puntos\_favor | smallint |  |  | Número de puntos conseguidos en total |
| puntos\_contra | smallint |  |  | Número de puntos recibidos en total |

Proba para CA7.11

* 1.- Crear e executar un procedemento almacenado que utilice un cursor para actualizar a táboa *clasificacion* utilizando a información da táboa *estatisticos*. A actualización empezará borrando as filas que teña a táboa *clasificacion*, e continuará inserindo nela información dos equipos ordenados polo número de partidos ganados (de maior a menor). No caso de empate, ordenarase pola diferenza entre puntos a favor e puntos en contra (de maior a menor). No caso de novo empate, non se seguirá ningún novo criterio para ordenar.

Exemplo de solución para entregar en papel

* Solución ao apartado 1
* Código do procedemento almacenado que utiliza un cursor

/\*

procedemento\_obter\_clasificacion\_cursor.sql

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

NOME RUTINA: liga\_acb.obter\_clasificacion\_cursor (procedemento con cursor)

DATA CREACIÓN: 25/11/2015

AUTOR: Grupo licenza 2015

TAREFA A AUTOMATIZAR: - Crear un procedemento almacenado con cursor para

actualizar a táboa clasificacion utilizando a información

da táboa estatisticos. A actualización empezará borrando

as filas que teña a táboa clasificacion, e continuará

inserindo nela información dos equipos ordenados polo

número de partidos ganados (de maior a menor). No caso de

empate, ordenarase pola diferenza entre puntos a favor e

puntos en contra (de maior a menor). No caso de novo

empate, non se seguirá ningún novo criterio para ordenar.

PARAMETROS REQUERIDOS: - Non existen.

RESULTADOS PRODUCIDOS: - Actualización da táboa clasificacion.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*/

delimiter **//**

**create** **procedure** liga\_acb**.**obter\_clasificacion\_cursor**()**

**begin**

**declare** vFinal **integer** **default** 0**;**

**declare** vPosicion **tinyint** **default** 0**;**

**declare** vEquipo **char(**4**);**

**declare** vXogados**,** vGanados **tinyint;**

**declare** vFavor**,** vContra **smallint;**

**declare** cursorClasificacion **cursor** **for**

**select**

equipo**,**

xogados\_local**+**xogados\_visitante**,**

ganados\_local**+**ganados\_visitante **as** ganados**,**

puntos\_favor\_local**+**puntos\_favor\_visitante **as** puntos\_favor**,**

puntos\_contra\_local**+**puntos\_contra\_visitante **as** puntos\_contra

**from** liga\_acb**.**estatistica

**order** **by** ganados **desc,** puntos\_favor **-** puntos\_contra **desc;**

**declare** **continue** handler **for** **not** **found** **set** vFinal**=**1**;**

**delete** **from** liga\_acb**.**clasificacion**;**

**open** cursorClasificacion**;**

**repeat**

**fetch** cursorClasificacion **into** vEquipo**,** vXogados**,** vGanados**,** vFavor**,** vContra**;**

**if** vFinal **=** 0 **then**

**set** vPosicion **=** vPosicion**+**1**;**

**insert** **into** liga\_acb**.**clasificacion

**(**posicion**,** equipo**,** xogados**,** ganados**,** puntos\_favor**,** puntos\_contra**)**

**values**

**(**vPosicion**,** vEquipo**,** vXogados**,** vGanados**,** vFavor**,** vContra**);**

**end** **if;**

until vFinal **=** 1 **end** **repeat;**

**close** cursorClasificacion**;**

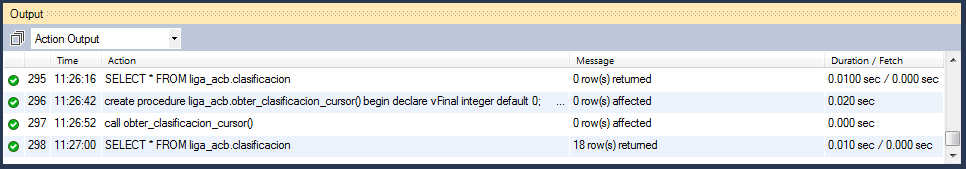
**end**

**//**

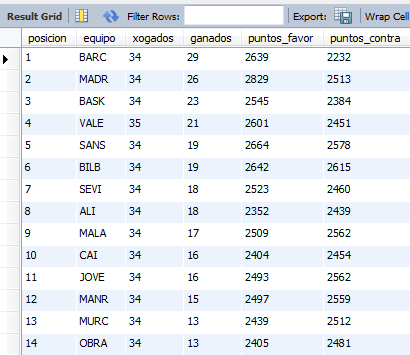
delimiter **;**

* Proba de funcionamento

Operacións realizadas:



Parte inicial do resultado na táboa *clasificacion*:



Exemplo de lista de valoración para TO.8

Proponse a seguinte lista de valoración para o instrumento de avaliación TO.8 (EV) (observación indirecta: documento de rexistro de creación e utilización de cursores).

| Nome | Data | |
| --- | --- | --- |
| Indicadores de  CA7.11 Utilizáronse cursores. | Valoración máxima  12 | Cualificación |
| Escribiuse o código correspondente ao cursor dentro do procedemento almacenado solicitado no exame. | 10 |  |
| Executouse o procedemento almacenado e mostrouse o resultado mediante imaxes capturadas en Workbench. | 2 |  |