Fundamentos de Bases de Datos

Práctica 2

Grosso Christian, Zhan Jiabo

DESCRIPCION GENERAL

En esta práctica hemos experimentado con el acceso a bases de datos desde un programa escrito en lenguaje C. En particular, hemos usado la librería ODBC para crear un programa que actúa sobre la base de datos ***classicmodels*** que hemos proporcionado en la primera práctica.

El programa se desarrolla en 4 archivos; el principal es **menu.c**, que contiene el "main" del programa y gestiona principalmente el menú principal. Los otros tres archivos, integrados como librerías .h, son **orders.c**, **customers.c** y **products.c**, que contienen las funciones necesarias para interactuar con la base de datos.

El menú principal tiene cuatro opciones: **Products**, **Orders**, **Customers**, y **Exit**. Mientras el usuario no seleccione la opción "Exit", el programa sigue ejecutándose y permite al usuario elegir entre las otras opciones del menú. Cada opción tiene su propio submenú con diferentes funcionalidades, y la lógica de cada submenú se implementa mediante consultas SQL específicas para cada funcionalidad.

1 - Menú Principal

- Este menú solicita al usuario seleccionar una opción entre **Products**, **Orders**, **Customers**, y **Exit** donde cada opción lleva a un submenú específico con funcionalidades propias y si se elige **Exit**, el programa se termina.
- Este archivo se compone de un main donde se gestiona el switch para los diferentes casos del menú y de una función que muestra el menú en pantalla.

2 - Submenú Products

Este archivo se compone de 3 funciones principales: la primera, que da la posibilidad de acceder a la consulta “Stock”, la segunda que da la posibilidad de acceder a la consulta “Find” y la tercera que permite de regresar al menú principal.

- **Stock**: Se solicita un productcode al usuario y se ejecuta una consulta SQL para obtener el número de unidades en stock del producto especificado.
  - ```
    SELECT quantityinstock
    FROM   products
    WHERE  productcode = ?
    ```

  En esta consulta se prevé la inserción de un parámetro que se solicita al usuario como entrada y se introduce dentro de la consulta con la función **SQLBindParameter**.

- **Find**: Se solicita una cadena de texto que puede ser parte del nombre de un producto. La consulta SQL busca en la base de datos todos los productos que contengan la cadena especificada en el nombre, mostrando el productcode y productname. El resultado se ordena por productcode.
  - ```
    SELECT productcode, productname
    FROM   products
    WHERE  productname LIKE ?
    ```

  También en esta consulta se prevé la inserción de un parámetro. Además, se utiliza LIKE para que sea posible introducir solo una parte del nombre del producto y no necesariamente el nombre exacto.

- **Back**: Permite al usuario regresar al menú principal.

3 - Submenú Orders

Este archivo se compone de 4 funciones principales: la primera, que da la posibilidad de acceder a la consulta “Open”, la segunda que da la posibilidad de acceder a la consulta “Range”, la tercera que da la posibilidad de acceder a la consulta “Detail”, y la cuarta que permite de regresar al menú principal.

- **Open**: Devuelve una lista de todos los pedidos que aún no han sido enviados (shippeddate no asignada). La salida muestra ordernumber y está ordenada por este atributo.

  ```
  SELECT ordernumber
  FROM   orders
  WHERE  shippeddate IS NULL
  ORDER  BY ordernumber
  ```

- **Range**: Se solicita un rango de fechas (formato YYYY-MM-DD). La consulta SQL devuelve los pedidos realizados en ese intervalo, incluyendo ordernumber, orderdate, y shippeddate. Los resultados se ordenan por ordernumber.

  ```
  SELECT ordernumber, orderdate, shippeddate
  FROM   orders
  WHERE  orderdate >= ? AND orderdate <= ?
  ```

  En esta consulta se prevé la inserción de dos parámetros que se solicitan al usuario como entrada y se introducen dentro de la consulta con la función **SQLBindParameter**.

- **Detail**: Se solicita un ordernumber. La consulta devuelve los detalles del pedido: la orderdate, el estado (si ha sido enviado o no), el coste total, y una lista de productos en el pedido (incluyendo productcode, quantityordered, priceeach). Las líneas de productos se ordenan por orderlinenumber.

  ```
  SELECT ordernumber, orderdate, status, productcode,
  quantityordered, priceeach, orderlinenumber
  FROM   orders
         natural JOIN orderdetails
  WHERE  ordernumber = ?
  ORDER  BY orderlinenumber
  ```

- **Back**: Regresa al menú principal.

4 - Submenú Customers

Este archivo se compone de 4 funciones principales: la primera, que da la posibilidad de acceder a la consulta "Find", la segunda que da la posibilidad de acceder a la consulta "List Products", la tercera que da la posibilidad de acceder a la consulta "Balance", y la cuarta que permite de regresar al menú principal.

- **Find**: Solicita una cadena de texto y devuelve un listado de clientes cuyos nombres o apellidos de contacto contengan esa cadena. Muestra el customername, contactfirstname, contactlastname, y el customernumber. La lista está ordenada por customernumber.

  ```
  SELECT customername, contactfirstname, contactlastname,
         customernumber
  FROM   customers
  WHERE  contactfirstname LIKE ?
          OR contactlastname LIKE ?
  ORDER  BY customernumber
  ```

  También en esta consulta se prevé la inserción de un parámetro. Además, se utiliza LIKE para que sea posible introducir solo una parte del nombre del producto y no necesariamente el nombre exacto.

- **List Products**: Se solicita el customernumber y devuelve todos los productos solicitados por el cliente en cualquier pedido. La salida incluye el productname y el número total de unidades solicitadas. Los productos se agrupan y se ordenan por productcode.

  ```
  SELECT productname, productcode,
  SUM(quantityordered) AS total
  FROM   orders
         NATURAL join orderdetails
         NATURAL join products
  WHERE  customernumber = ?
  GROUP  BY productname,
            productcode
  ORDER  BY productcode ASC
  ```

- **Balance**: Solicita el customernumber y calcula el saldo del cliente. Para calcularlo, se realiza la diferencia entre la suma de todos los pagos realizados y el costo total de los productos comprados por el cliente.

  ```
  SELECT(SELECT Sum(amount) AS saldo
         FROM   payments
         WHERE  customernumber = ?) -
  (SELECT Sum(od.quantityordered * od.priceeach) AS buyed
   FROM   customers c
   natural JOIN orders
   natural JOIN orderdetails od
   WHERE  c.customernumber = ?) AS risultato
  ```

- **Back**: Regresa al menú principal.