Proyecto "microSQL"

Estructura de Datos I

Objetivos del Proyecto:

- Aplicar los conocimientos adquiridos durante el curso para el desarrollo de una aplicación de software.
- Codificar soluciones para un problema dado por medio de un enfoque Orientado a Objetos.
- Creación y manipulación de distintas estructuras de datos en C#.
- Aplicación de las mejores prácticas de desarrollo vistas en clase.

Organización:

- Parejas o Individual

Entregables:

- **Fecha primera entrega: Jueves 06/04/2017 03:00 horas.**
- Documento de Análisis y Diseño de Software. El diseño de bajo nivel debe modelar todos los procesos identificados.
- **Fecha de entrega final: Jueves 27/04/2017 03:00 horas.**
- Código fuente del programa, el archivo ejecutable y la documentación interna del código fuente.
- Manual de usuario, el formato del mismo queda a decisión del estudiante (documento escrito, video tutorial, animación, etc.).
- Todos estos entregables deberán ser subidos al portal del curso empaquetados dentro de un archivo "*.rar" con los nombres y carné de los alumnos.

Aspecto a evaluar:

- Adecuada aplicación de los conocimientos.
- Calidad de la documentación: ortografía, orden, limpieza y que esté completa (diagramas UML para representar el diseño de las clases y diagramas de flujo para representar el orden de ejecución del programa).

- Funcionalidad del programa: debe cumplir a cabalidad con todos los requerimientos especificados en este documento.
- Evidencia de la creación del programa y dominio de los conceptos utilizados. Se podrá demandar que en la calificación presencial se realicen cambios de alguna funcionalidad, así como revisión de la bitácora del controlador de versiones.
- Creatividad.
- Utilizar técnicas de ESTILO de programación (documentación interna, convenciones de nombres, etc.)

DESCRIPCIÓN DEL PROYECTO

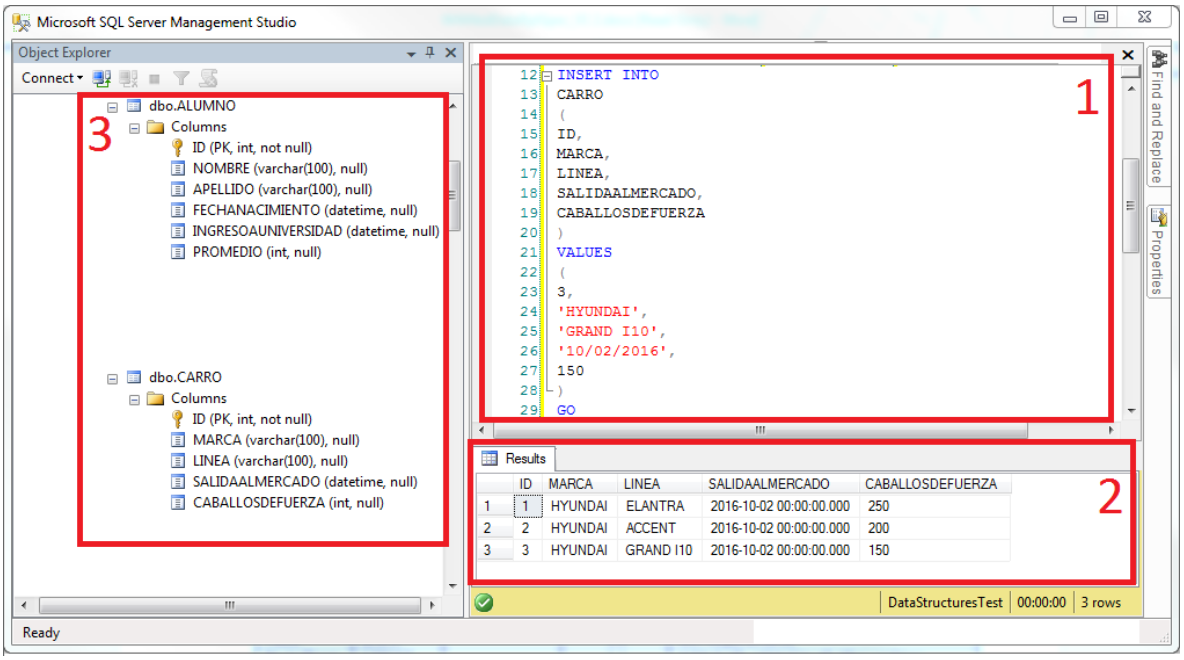
Actualmente usted es el director del proyecto de desarrollo de “microSQL”, el más reciente producto de la empresa “Soluciones Tecnológicas S.A.” (STSA). El producto es un manejador de base de datos (DBMS) que le permita gestionar sus estructuras de datos de forma personalizada, pudiéndose acoplar a varios idiomas distintos del Inglés, lo que permitirá la adopción de esta herramienta por una mayor cantidad de usuarios.

Debido a que su equipo se encontraba atrasado con respecto a la planificación, usted tercerizó el desarrollo del componente ***Estructuras De Datos.dll*** el cual le estará entregando durante las próximas semanas (misma DLL que se estará trabajando durante los laboratorios del curso). Este componente contendrá todo lo relacionado al manejo de las estructuras de datos genéricas Árbol B, Árbol Binario De Búsqueda y Árbol AVL.

El programa deberá de mostrar una interfaz gráfica con tres secciones:

1. una primera sección donde se pueda introducir código SQL para manipular los objetos de la base de datos
2. otra sección deberá incluir un “grid” donde pueda visualizarse el resultado de las instrucciones de recuperación de datos ejecutadas
3. una tercera dónde se pueda visualizar de forma atractiva el listado de tablas creadas y sus respectivas columnas.

En la sección de código SQL, se pueden aplicar varias instrucciones en una sola ejecución, siempre y cuando estas instrucciones se encuentren separadas por una línea con la palabra GO.



MicroSQL debe permitir la gestión de la base de datos tomando en cuenta los siguientes puntos:

a. Debido a que el programa debe aceptar comandos en varios idiomas, se pueden sustituir las palabras reservadas del lenguaje SQL, por lo que al iniciar el programa se debe cargar un diccionario de palabras reservadas desde el archivo ***C:/microSQL/microSQL.ini***. Dicho archivo consiste en un archivo CSV que contiene por cada línea una pareja de palabras de la siguiente forma:

<Palabra Reservada>, <Comando en otro idioma>

Palabra Reservada (esta parte no debe cambiar)	Comando en otro idioma (esta es la parte que sí cambia, abajo se presentan los valores por defecto)
SELECT	SELECT
FROM	FROM
DELETE	DELETE
WHERE	WHERE

CREATE TABLE	CREATE TABLE
DROP TABLE	DROP TABLE
INSERT INTO	INSERT INTO
VALUES	VALUES
GO	GO

Si el usuario desea personalizar este diccionario puede hacerlo mediante la sustitución del archivo por uno nuevo que contenga la nueva definición. Si no existiera dicho archivo, el sistema deberá generar uno nuevo colocando los valores por defecto.

Así mismo la configuración de la aplicación también debe permitir la carga de una nueva definición del diccionario desde la interfaz gráfica, así como una opción que permita restablecer a los valores predeterminados.

Estas palabras reservadas deberán ser mostradas en un color distinto en la interfaz gráfica en la sección para introducción de código SQL.

b. El sistema debe permitir las siguientes funcionalidades:

- Creación de nueva tabla:
  - Permite la creación de una nueva tabla en la base de datos (internamente será un árbol B).
  - Los nombres de las columnas no pueden contener espacios.
  - Los tipos de columnas admitidas son: INT, VARCHAR(100) y DATETIME.
  - El máximo número de columnas por cada tipo de dato es 3.
  - Se deberá agregar obligatoriamente una columna de nombre ID de tipo INT que actuará cómo llave primaria.
  - Se debe validar que la sintaxis cumpla con todas la reglas mencionadas, de lo contrario deberá mostrarse un error en pantalla.
  - Este proceso deberá crear un archivo denominado ***<nombre de la tabla>.arbolb*** dentro de la carpeta

C:/microSQL/arbolesb/, dicho archivo contendrá los datos que se ingresen en la tabla, es decir, es el árbol B.

- Adicionalmente el proceso deberá crear un archivo denominado ***<nombre de la tabla>.tabla*** dentro de la carpeta ***C:/microSQL/tablas/*** en el que deberá guardar la estructura de la tabla (nombres de las columnas de la tabla y tipos de datos) los cuales serán utilizados por otras funcionalidades.

Sintaxis:

```
CREATE TABLE
<NOMBRE DE LA TABLA>
(
ID INT PRIMARY KEY,
<NOMBRE DE LA COLUMNA> <TIPO DE DATO DE LA COLUMNA>,
...
)
```

```
CREATE TABLE
CARRO
(
ID INT PRIMARY KEY,
MARCA VARCHAR(100),
LINEA VARCHAR(100),
SALIDAALMERCADO DATETIME,
CABALLOSDEFUERZA INT
)
```

- Inserción de datos:

  - Permite el ingreso de datos a una tabla específica de la base de datos.
  - Se debe validar que la sintaxis cumpla con todas la reglas mencionadas, de lo contrario deberá mostrarse un error en pantalla.
  - Las columnas/datos para la inserción siempre se proporcionarán en el mismo orden en que fueron indicados en la creación de la tabla.

- Se proporcionarán valores para todos los campos ya que no se permiten campos “Nulos”.
- Los valores de campos VARCHAR deberán delimitarse por medio del símbolo ‘ (apóstrofo o comilla simple).

Sintaxis:

```
INSERT INTO
<NOMBRE DE LA TABLA>
(
ID,
<NOMBRE DE LA COLUMNA>,
...
)
VALUES
(
<VALOR A INGRESAR>,
...
)
```

```
INSERT INTO
CARRO
(
ID,
MARCA,
LINEA,
SALIDAALMERCADO,
CABALLOSDEFUERZA
)
VALUES
(
1,
'HYUNDAI',
'ELANTRA',
'10/02/2016',
250
)
```

- Selección de datos:
    - Permite la extracción de datos de la Base de Datos, para una tabla específica.
    - Los datos extraídos deben mostrarse en el grid previamente mencionado.
    - El contenido del grid se deben poder exportar a un archivo CSV escogido por el usuario.
    - Se debe validar que la sintaxis cumpla con todas la reglas mencionadas, de lo contrario deberá mostrarse un error en pantalla.
    - Si los campos indicados en la consulta no existen deberá mostrarse un error en pantalla indicando cuales son los campos inexistentes en la tabla.

Sintaxis:

Seleccionar todos los datos:

```
SELECT
<NOMBRE DE LA COLUMNA>,
...
FROM
<NOMBRE DE LA TABLA>
```

```
SELECT
ID,
LINEA,
MARCA
FROM
CARRO
```

```
SELECT
*
FROM
<NOMBRE DE LA TABLA>
```

```
SELECT
*
FROM
CARRO
```

Agregar filtro a la llave primaria:

```
SELECT
<NOMBRE DE LA COLUMNA>,
...
FROM
<NOMBRE DE LA TABLA>
WHERE
ID = <VALOR A BUSCAR>
```

```
SELECT
ID,
LINEA,
MARCA
FROM
CARRO
WHERE
ID = 1
```

```
SELECT
*
FROM
<NOMBRE DE LA TABLA>
WHERE
ID = <VALOR A BUSCAR>
```

```
SELECT
*
FROM
CARRO
WHERE
ID = 1
```

Nota:
El * indica que se deben mostrar todas las columnas de la tabla, es el equivalente a escribir el nombre de todos los campos.

- Eliminación de datos:
  - Permite eliminar datos almacenados en una tabla (eliminar filas de la tabla).
  - Se debe validar que la sintaxis cumpla con todas la reglas mencionadas, de lo contrario deberá mostrarse un error en pantalla.

Sintaxis:

Elimina todos los datos:

```
DELETE FROM
<NOMBRE DE LA TABLA>
```

```
DELETE FROM
CARRO
```

Elimina según filtro de llave primaria:

```
DELETE FROM
<NOMBRE DE LA TABLA>
WHERE
ID = <VALOR A BUSCAR>
```

```
DELETE FROM
CARRO
WHERE
ID = 1
```

- Eliminación de tablas:
  - Elimina toda la tabla y sus datos, se deberán borrar todos los archivos asociados.
  - Se debe validar que la sintaxis cumpla con todas la reglas mencionadas, de lo contrario deberá mostrarse un error en pantalla.

Sintaxis:

```
DROP TABLE
<NOMBRE DE LA TABLA>
```

```
DROP TABLE
CARRO
```

Requerimientos extras:

- Procesar las instrucciones SQL sin importar el espaciado o saltos de línea encontrados en las instrucciones.
- Que el Grid pueda exportarse a formato de Excel XLSX, adicionalmente a la opción para exportarlo a CSV.
- Aplicar el filtrado de las instrucciones SELECT y DELETE a columnas distintas de ID.
- Implementación de la opción para actualizar datos presentes en las tablas, instrucción UPDATE.

```
UPDATE
< nombre de la tabla>
SET
< nombre de la columna > = < valor >
WHERE
ID = < valor >
```