Proyecto Organización de Archivos

Alumnos:
Joel Pineda
Josue Rojas

Catedrático:
Ing. Carlos Arias

Fecha:
23 de Septiembre
de 2013

Introducción

En el mundo actual la demanda de información digital es bastante alta. Para satisfacer esta necesidad es preciso poder almacenar grandes cantidades de datos que se puedan acceder casi de manera inmediata. Pero existe un problema ya que los dispositivos que almacenan grandes cantidades de datos tienen una transferencia demasiada lenta en comparación con la memoria principal de las computadoras. Es por ello que en la clase de Organización de Archivos hemos aprendido las técnicas necesarias para poder hacer programas que manejen grandes cantidades de información en el disco duro y sean bastante eficientes.

A continuación se presenta nuestro proyecto, en el cual se implementan los conocimientos obtenidos mediante el curso de Organización de Archivos.

Marco Teórico

Marco Teórico

Nuestro proyecto consiste en un programa desarrollado en C++, el cual tiene como función administrar archivos de registro que pueden contener diferentes tipos de información.

A continuación se explican detalladamente algunos términos necesarios para poder comprender de una mejor manera el comportamiento del mismo:

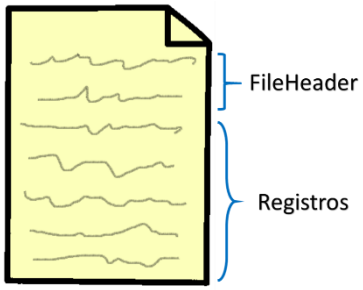
Campos y Llaves:
Un campo es un espacio en donde se almacena un dato en específico, es la unidad más pequeña dentro de una tabla de registros. Estos pueden contener diferentes tipos de información como ser enteros, cadenas, caracteres, entre otros.
Los campos pueden también ser Llaves Primarias o Llaves Secundarias, estas sirven para identificar a un registro. Las llaves primarias identifican de manera única a un registro y no se pueden repetir, mientras que las llaves secundarias pueden identificar a más de un registro.

Registro:
Es un conjunto de campos agrupados que guardan información completa sobre una entidad, persona o cosa. Los registros pueden ser de un tamaño fijo o de un tamaño variable.

Archivos de Registro:
Son archivos que se pueden almacenar y leer desde o hacia un dispositivo de almacenamiento como ser un disco duro o una memoria USB. Estos en su interior contienen registros y campos.

File Header:

Es la parte de un archivo de registros que puede estar al principio o al final de este. En el File Header se encuentra la información necesaria para poder cargar todos los registros del archivo.



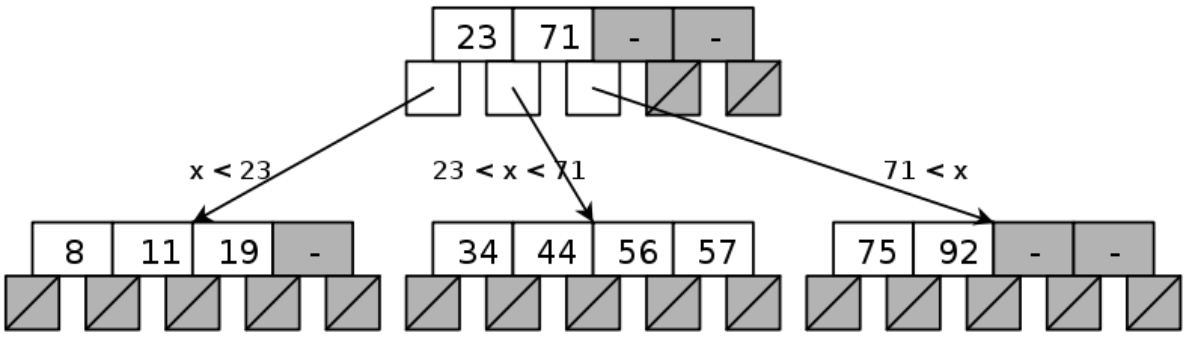
Índice:

Es un archivo externo al archivo de registros que se utiliza para agilizar el proceso de lectura/escritura, este contiene las llaves de manera ordena y su correspondiente referencia hacia el registro en el archivo principal, esto ayuda a mejorar notablemente el rendimiento.

Árbol B:

Son estructuras de datos de tipo árbol que comúnmente se utilizan en las bases de datos y sistemas de administración de archivos para mejorar su rendimiento, ya que estos árboles disminuyen drásticamente el tiempo que se requiere para obtener un registro de un archivo.

Son arboles balanceados de búsqueda en los que sus nodos pueden contener más de dos hijos, además garantizan que sus datos estén siempre ordenados y que la inserción y eliminación se dé un tiempo algorítmico.

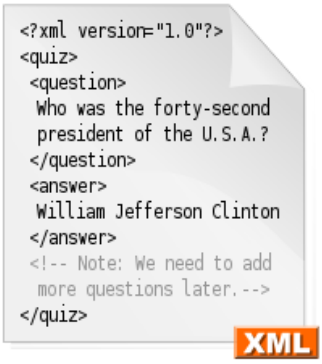


QT:

Es una biblioteca de código multiplataforma que se utiliza para el desarrollo de aplicaciones con interfaz gráfica, así como también para el desarrollo de programas sin interfaz gráfica. Cabe recalcar que Qt utiliza el lenguaje de programación C++ de forma nativa y que es desarrollada como un software libre y de código abierto a través de Qt Project.
Qt ha sido implementado en el desarrollo de nuestro proyecto.
QT se puede descargar de la siguiente página: http://qt-project.org/downloads.

XML:

El lenguaje de marcas extensible (XML) es utilizado para guardar datos en forma legible. Al igual que HTML, XML está estructurado mediante etiquetas y sirve para documentos grandes.
Un punto muy importante es que XML da soporte a bases de datos y otros lenguajes no, siendo esto útil cuando varias aplicaciones que se deben comunicar entre sí o integrar información.



JSON (JavaScript Object Notation):

Según Wikipedia, JSON es un formato ligero para el intercambio de datos, también es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML.
El uso de JSON se ha generalizado debido a su simplicidad, especialmente como alternativa a XML en AJAX.

Github:

Es un servicio de alojamiento de repositorios de software muy popular entre los programadores que permite alojar código de forma gratuita siempre y cuando este sea visible para todas las demás personas de la comunidad, también cuenta con versiones de pago que permiten crear repositorios privados.

Una de las grandes características de Github es que permite muy fácilmente programar de una manera colaborativa entre varias personas en cualquier parte del mundo.

Actualmente grandes empresas como Facebook alojan en Github sus desarrollos públicos.

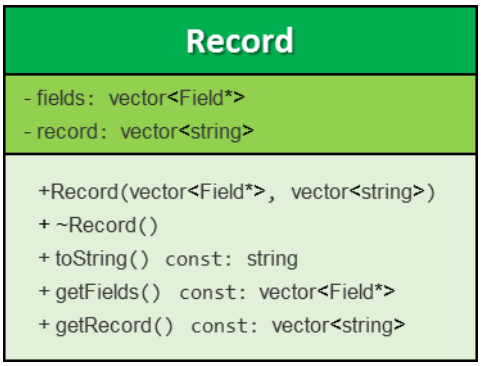
Expresiones Regulares:

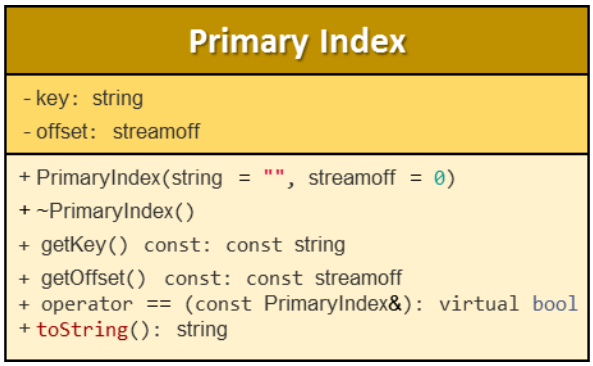
Una expresión regular o regex es un conjunto de caracteres que forman un patrón de búsqueda. Generalmente se usa para buscar patrones de cadenas de caracteres. En conclusión las Expresiones Regulares son un lenguaje que nos permite definir exactamente qué es lo que queremos buscar o reemplazar dentro de un texto. Los caracteres especiales de las expresiones regulares son conocidos como metacaracteres.

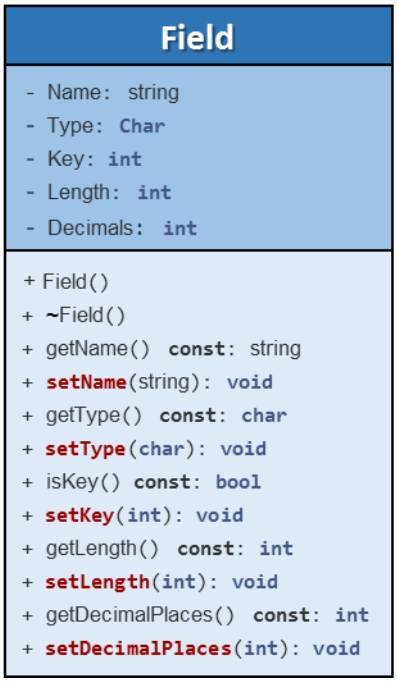
Ejemplo:

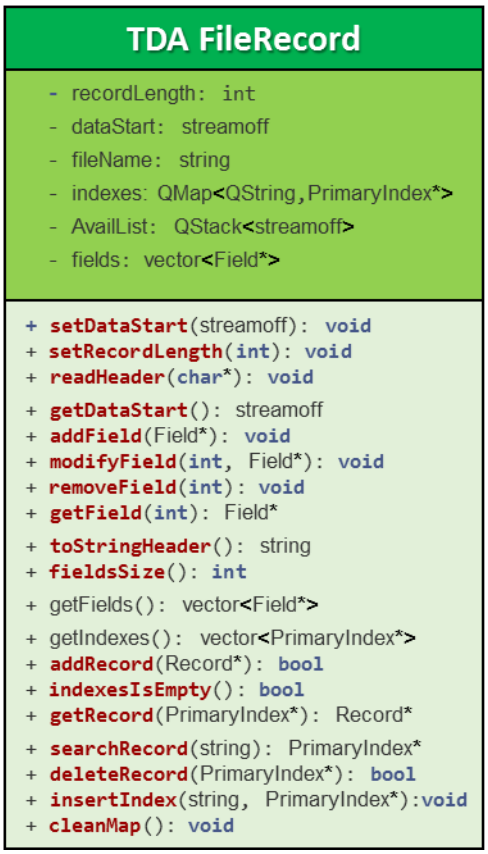
Ejemplos de regex	(\W|^)(tontería|maldito|caray|caramba|madre\smía|ostras)(\W|$)
Notas	• (...) agrupa todas las palabras de modo que la clase de carácter \W afecta a todas las palabras del paréntesis. • \W coincide con cualquier carácter que no sea una letra, un número o un guión bajo. Impide que la expresión regex coincida con caracteres que precedan o sigan a las palabras o a las frases de la lista. • ^ coincide con el inicio de una nueva línea. Permite a la expresión regex coincidir con la palabra si aparece al principio de la línea, sin ningún carácter que le preceda. • $ coincide con el final de una línea. Permite a la expresión regex coincidir con la palabra si aparece al final de la línea, sin ningún carácter que le siga. • | indica la disyuntiva “o,” para que la expresión regex coincida con cualquiera de las palabras de la lista. • \s coincide con un carácter de espacio. Utilízalo para separar las palabras en una frase.

Implementación


Record
- fields: vector<Field*>
- record: vector<string>
+Record(vector<Field*>, vector<string>)
+ ~Record()
+ toString() const: string
+ getFields() const: vector<Field*>
+ getRecord() const: vector<string>


Primary Index
- key: string
- offset: streamoff
+ PrimaryIndex(string = "", streamoff = 0)
+ ~PrimaryIndex()
+ getKey() const: const string
+ getOffset() const: const streamoff
+ operator == (const PrimaryIndex&): virtual bool
+ toString(): string


Field
- Name: string
- Type: Char
- Key: int
- Length: int
- Decimals: int
+ Field()
+ ~Field()
+ getName() const: string
+ setName(string): void
+ getType() const: char
+ setType(char): void
+ isKey() const: bool
+ setKey(int): void
+ getLength() const: int
+ setLength(int): void
+ getDecimalPlaces() const: int
+ setDecimalPlaces(int): void


TDA FileRecord
- recordLength: int
- dataStart: streamoff
- fileName: string
- indexes: QMap<QString,PrimaryIndex*>
- AvailList: QStack<streamoff>
- fields: vector<Field*>
+ setDataStart(streamoff): void
+ setRecordLength(int): void
+ readHeader(char*): void
+ getDataStart(): streamoff
+ addField(Field*): void
+ modifyField(int, Field*): void
+ removeField(int): void
+ getField(int): Field*
+ toStringHeader(): string
+ fieldsSize(): int
+ getFields(): vector<Field*>
+ getIndexes(): vector<PrimaryIndex*>
+ addRecord(Record*): bool
+ indexesIsEmpty(): bool
+ getRecord(PrimaryIndex*): Record*
+ searchRecord(string): PrimaryIndex*
+ deleteRecord(PrimaryIndex*): bool
+ insertIndex(string, PrimaryIndex*):void
+ cleanMap(): void

Implementación

Clase ADTFile:
Es una clase de tipo abstracta que contiene las funciones necesarias para el trabajo con archivos de diferente tipo. En el interior de ella se implementa la clase *fstream* de C++.
En la clase ADTFile se encuentran funciones como: Abrir y Cerrar un Archivo, escribir o leer, mover los punteros de lectura/escritura, entre otras funciones necesarias para el manejo correcto de los archivos en C++.

Clase ADTFileRecord:
Descendiente de ADTFile, contiene métodos extras necesarios para manipular los archivos de registro de una mejor manera. Esta clase tiene métodos especiales que ayudan por ejemplo a leer solamente el archivo deseado y a interpretar el Header del archivo de registros.

Clase Field:
Es una clase que contiene toda la información principal de un campo. Esta información es interpretada para luego servir de parámetro a los registros de todo el archivo.

Clase Object:
Esta clase ha sido creada con el fin de simular algo parecido al Object de Java. El objetivo es que se pueda emplear como un tipo dentro de otras clases para que podamos guardar cualquier objeto en ella.

Clase PrimaryIndex:
Clase que se utiliza para manejar los índices primarios de un archivo de registro, permiten una búsqueda rápida y son de mucha ayuda en la lectura de registros.

Clase Record:
Clase importante para la manipulación de los registros que están contenidos en el archivo, pues maneja dos vectores de la misma longitud que mantiene una correspondencia entra cada uno de elementos de cada vector.

Clase Main:
Es la clase encargada de manejar toda la lógica del programa, desde ella se levanta la parte gráfica y todas las demás funciones.

Muestra de Corrida

Muestra de Corrida

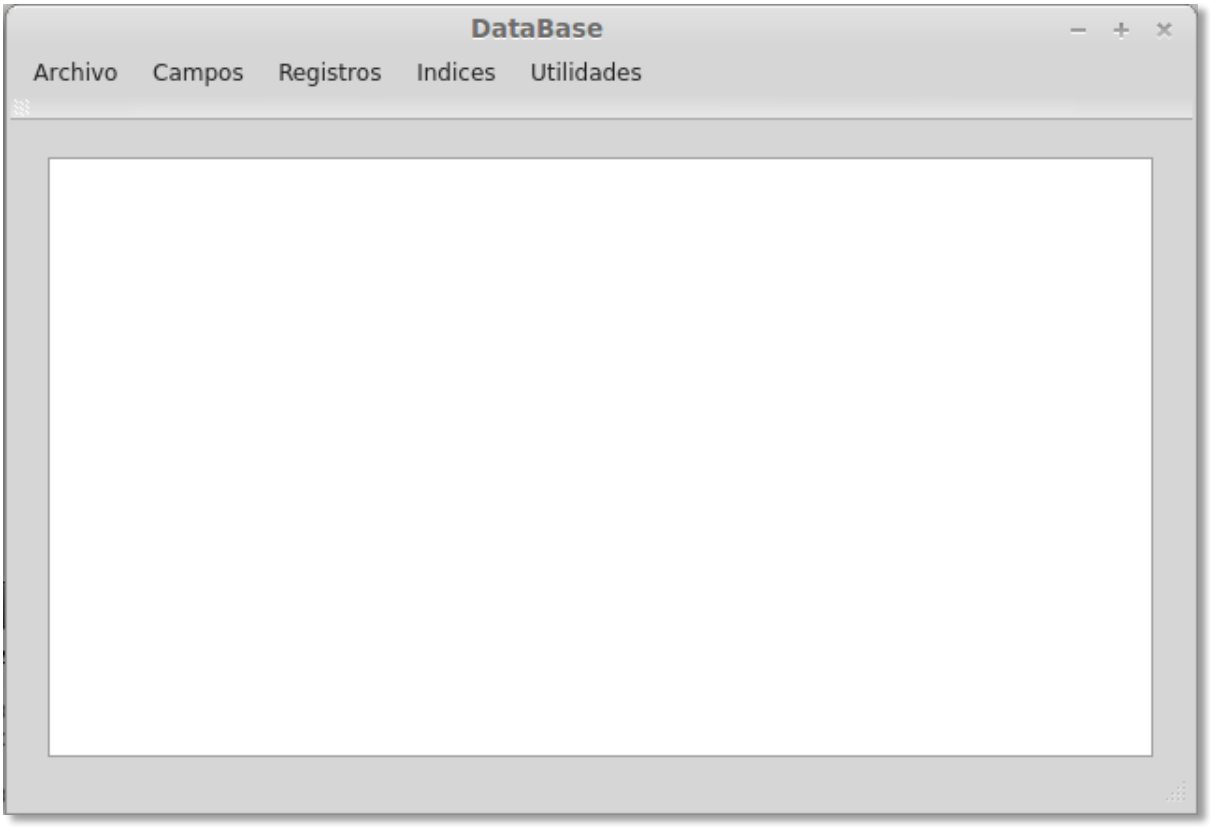


Imagen de la Ventana Principal

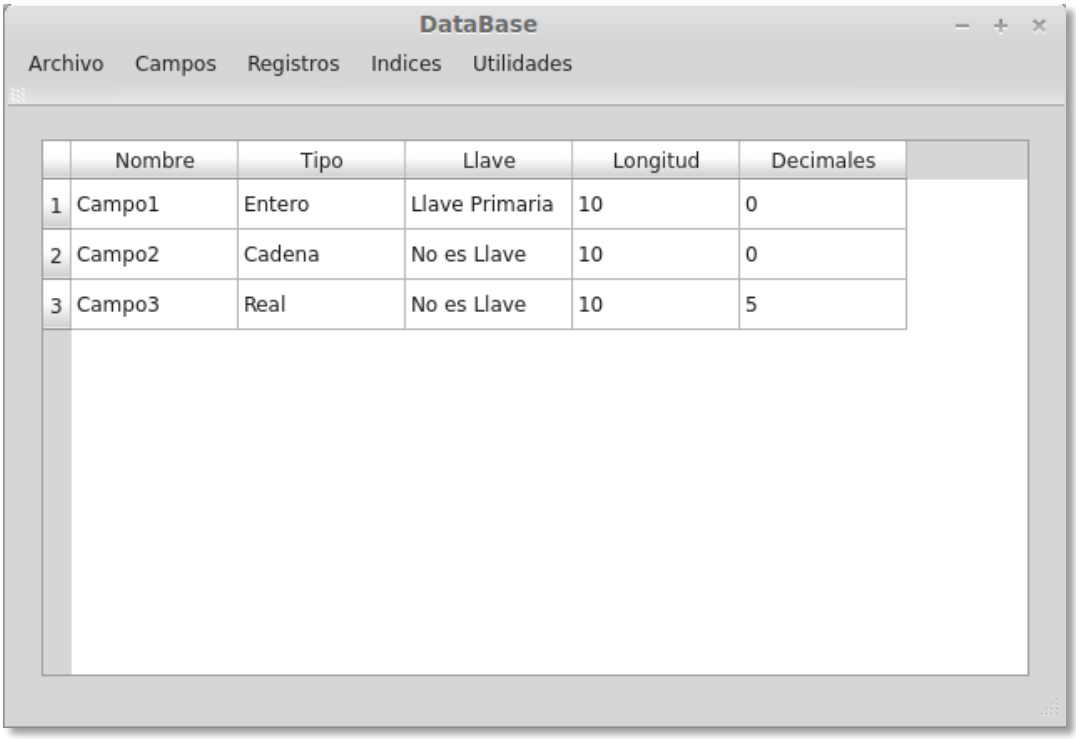
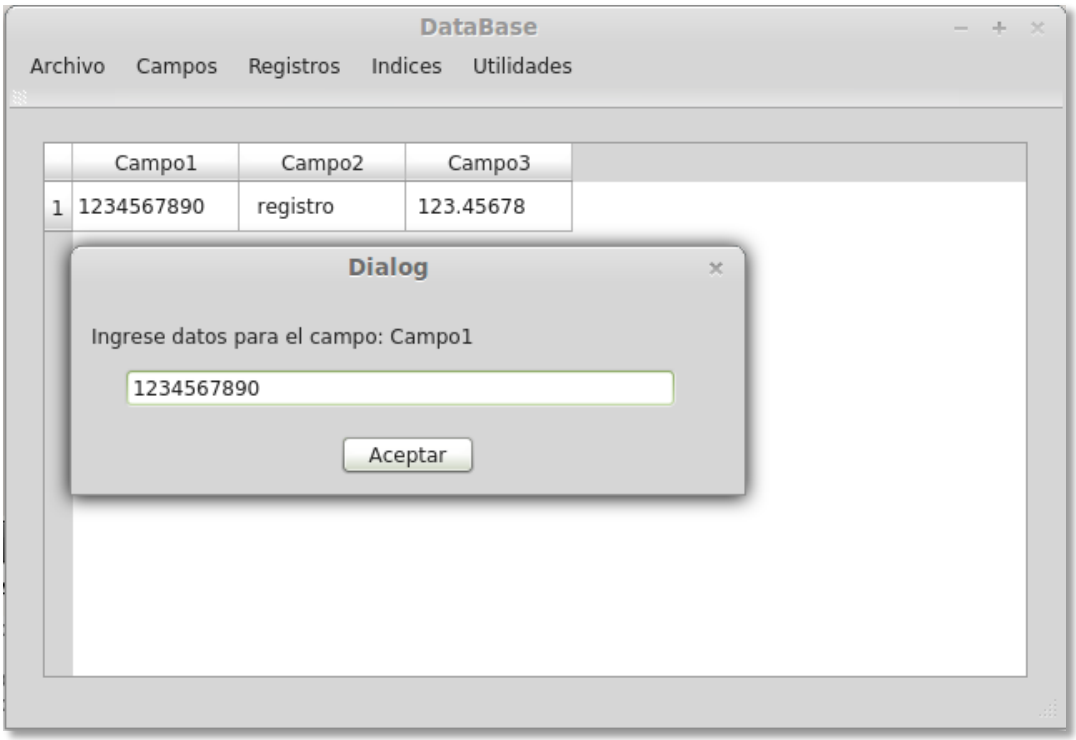


Ilustración del Programa al momento de listar los campos



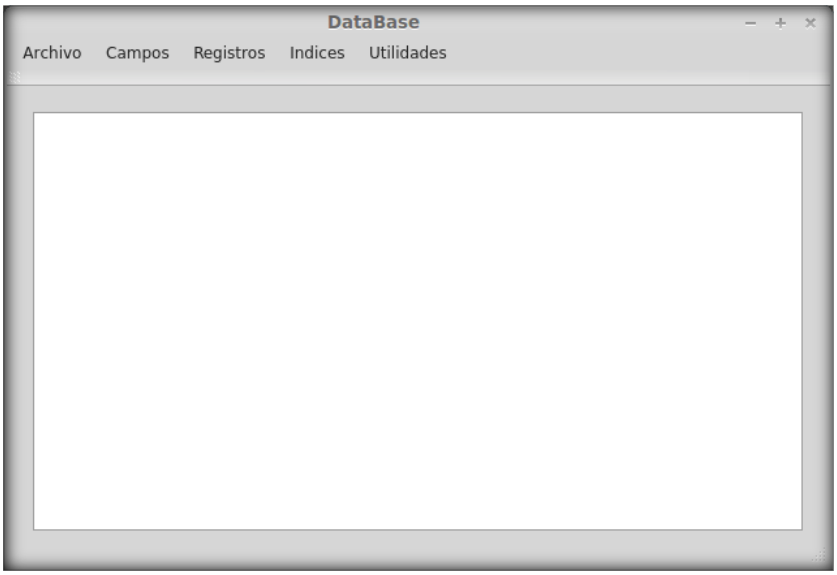
Ventana para buscar registros.

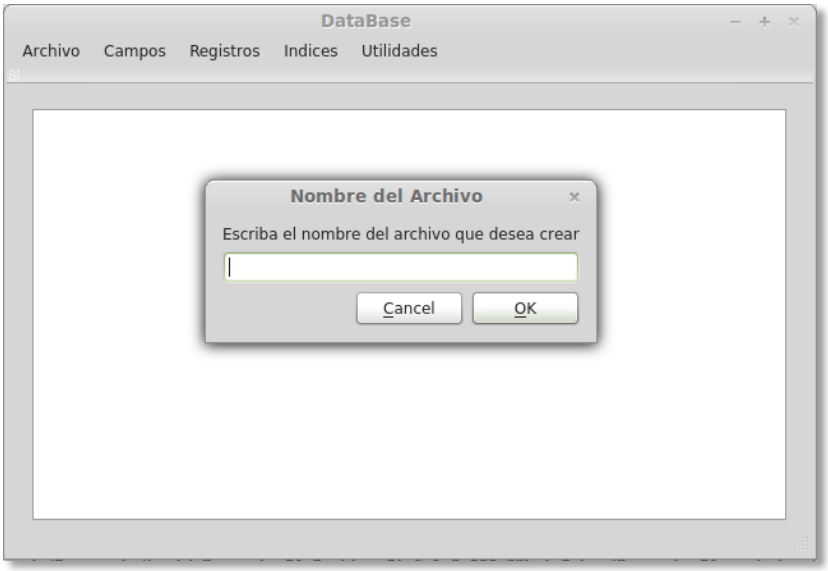
Manual De Usuario

Manual de Usuario

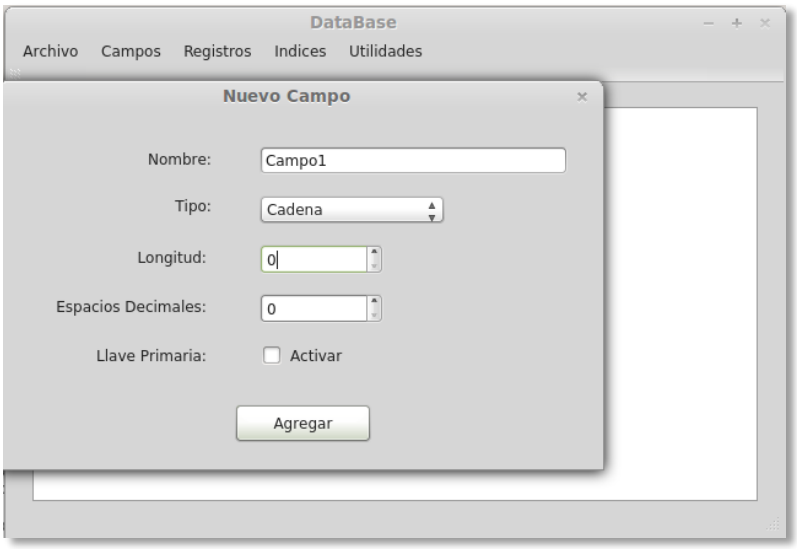
Al momento de ejecutar el programa se muestra esta pantalla principal.

En la parte superior podemos observar varias opciones que nos ayudarán y nos brindaran las herramientas necesarias.



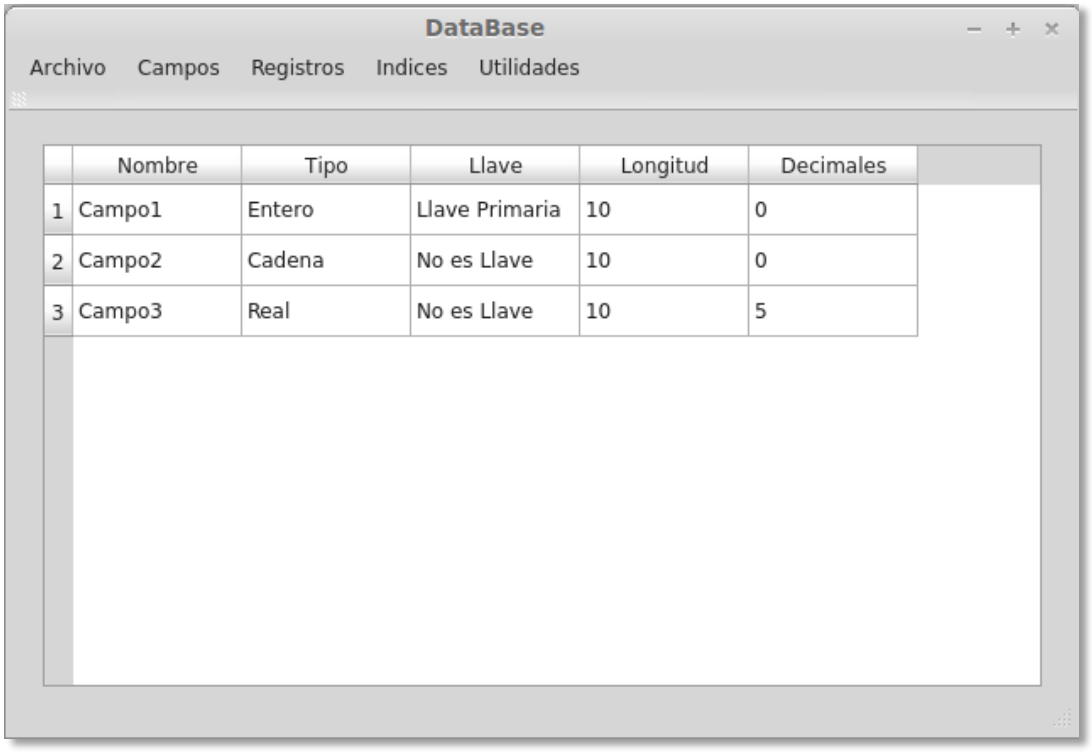


Antes que todo debemos de crear un archivo de registros que contendrá nuestros datos. Para ello, en el botón de “Archivo” se encuentra la opción para crear un nuevo archivo, el programa nos pedirá el nombre que le queremos dar al archivo.



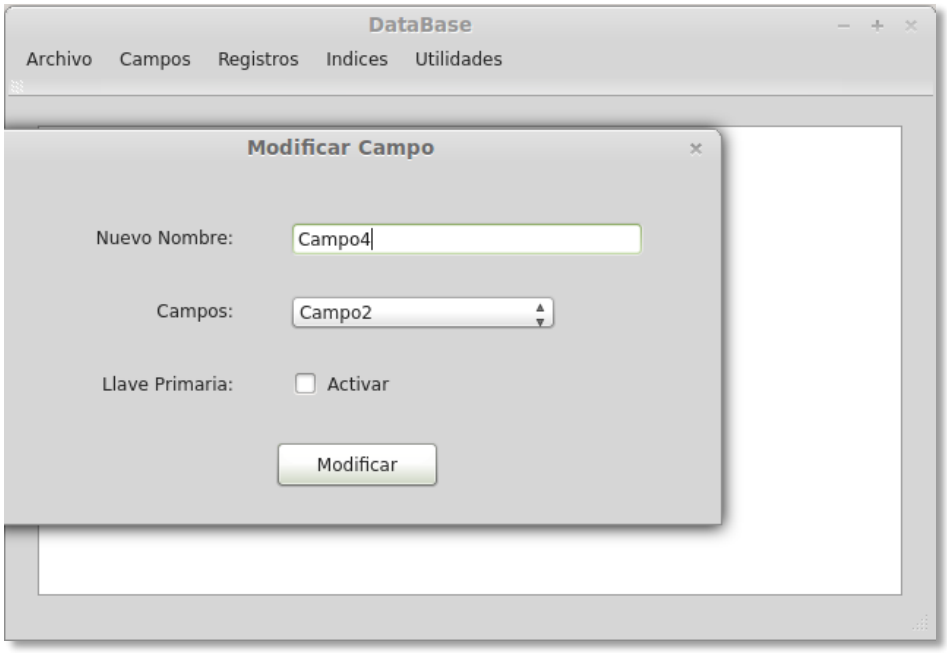
Después de crear un archivo de registro, debemos crear los campos que deseamos. Es necesario que el archivo por lo menos contenga un campo.

La ventana de nuevo campo le solicitará al usuario los datos necesarios para crear un campo.

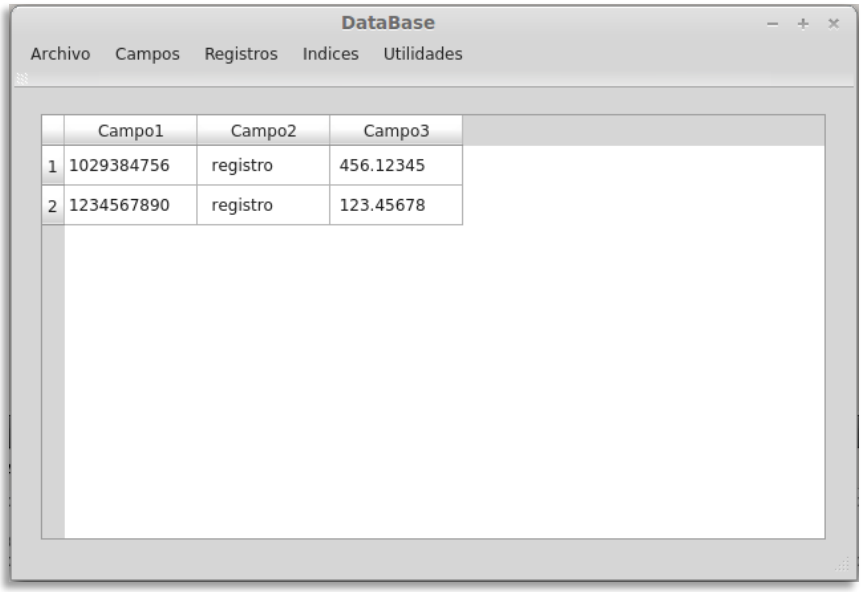


	Nombre	Tipo	Llave	Longitud	Decimales
1	Campo1	Entero	Llave Primaria	10	0
2	Campo2	Cadena	No es Llave	10	0
3	Campo3	Real	No es Llave	10	5

Una vez creados los campos es posible listarlos y ver sus atributos.



Una vez creados los campos es posible modificar ciertas características de estos.

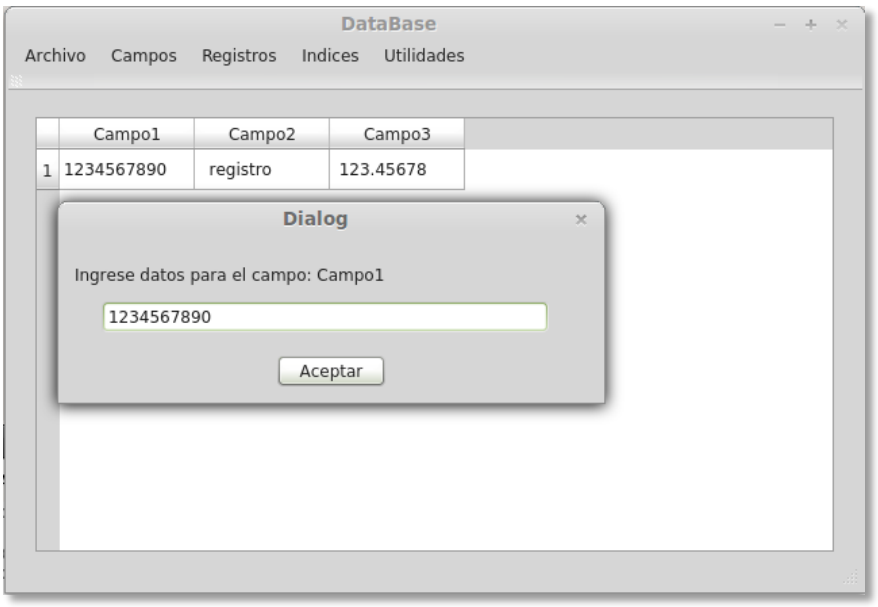


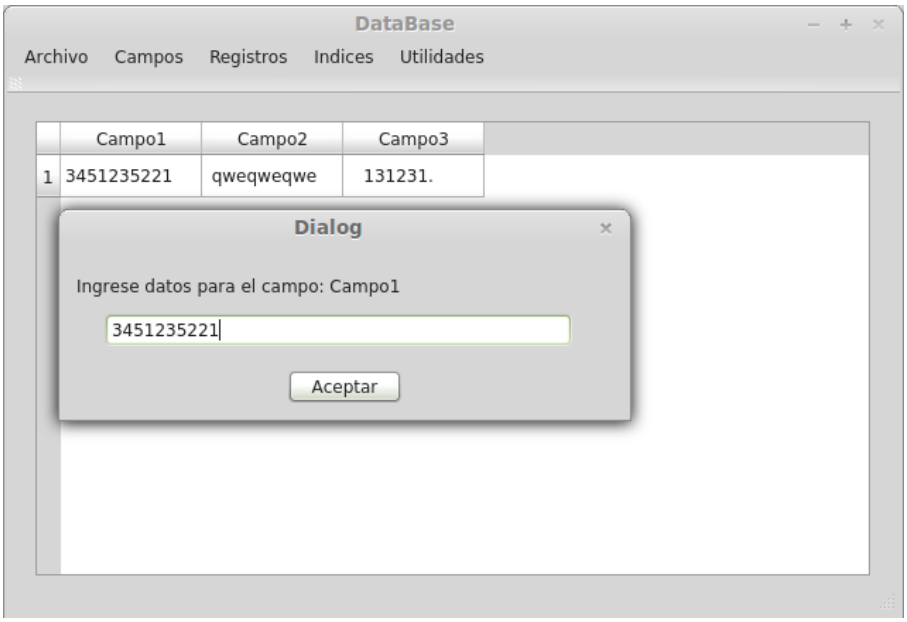
Si el usuario ya ha ingresado al menos un registro, es posible listarlos dentro de una tabla como la que se muestra en la imagen.

Para listar solamente se tiene que dar clic en el botón Registros y luego en Listar Registros.

Es posible realizar búsquedas dentro de los registros, solamente se debe de ingresar el campo llave del registro que se desea buscar.

Las búsquedas se realizan por medio de indices.

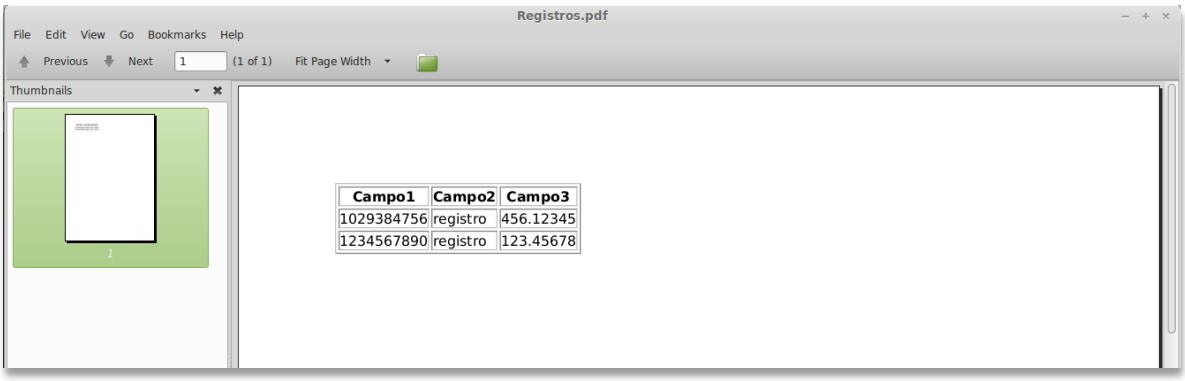




Eliminar Registros es muy fácil, solamente se debe de ingresar la llave correspondiente al registro que se quiere eliminar.

La opción eliminar registro está dentro de la ficha de registros.

Una de las características de este programa es que le permite al usuario exportar datos en formato PDF, algo que resulta muy útil si se desea imprimir una tabla o simplemente compartir los datos con las computadoras de otras personas.



También el programa permite exportar e importar archivos XML. Esta es una herramienta muy útil para compartir datos con otras bases de datos, ya que la mayoría de ellas soportan esta característica.

Experiencia con Github

El hecho de haber tenido que utilizar GitHub en nuestro proyecto nos ha ayudado a crecer mucho, ya que esta es una herramienta muy útil que nos servirá no solo durante estemos dentro de la universidad, sino también para un futuro en un ambiente laboral.

Herramientas como estas nos facilitan el trabajo en equipo y fomentan la programación colaborativa. En nuestro caso Github nos ayudó a trabajar desde nuestras casas e incluso desde diferentes ciudades, y sin esto se nos hubiera complicado demasiado unir nuestros avances y ver los avances hechos por nuestro compañero.

En cuanto a las dificultades con Github fueron muy pocas, solamente al principio el hecho de acostumbrarse a algo nuevo y de aprender los comandos básicos.

Creemos que los beneficios que nos brinda Github son muchos y que será muy útil seguirlo utilizando en nuestras futuras clases. Cabe mencionar también que contamos con repositorios privados de manera gratuita, ya que por el hecho de que seamos estudiantes ellos nos regalan ese servicio.

En conclusión Github es muy bueno y sus beneficios son muchos, lo que nos motiva a seguirlo utilizando, pero también sería bueno buscar y probar servicios parecidos a este de otras empresas para comparar sus características y elegir las mejores para explotarlas al máximo.

Conclusiones

Después del desarrollo del proyecto, en base a errores, soluciones y aspectos positivos podemos concluir lo siguiente:

- El manejo de archivos de gran tamaño en disco se puede convertir en un proceso ineficiente sino se utilizan las técnicas y estructuras de datos necesarias para reducir la cantidad de lecturas/escrituras en disco.

- QT es una gran herramienta, que tiene muchas ventajas como el hecho de ser multiplataforma, tener una interfaz gráfica muy bonita, tener una completa documentación y una comunidad de programadores de QT.

- El trabajo en equipo es un poco más complicado que el trabajo individual, quizás porque nos hemos acostumbrado a programar individualmente. Pero al final se pudo lograr un buen trabajo en equipo.

- Si de trabajar en equipo se trata, la mejor opción es optar por una herramienta como Github para mantener el código actualizado y al alcance de todos, así como también hacer uso de los Hangouts de Google+ para lograr una mejor comunicación, cabe recalcar que por medio de los Hangouts se pueden hacer llamadas, video llamadas, compartir pantallas y otras funciones útiles e interesantes.

- Al momento de validar la entrada de los datos, las expresiones regulares son bastante útiles, debido a que son muy fáciles de usar y funcionan bastante bien.