**Proyecto General: Data Fusion DB**

**Proyecto 1: Organización de Archivos**

El nombre DataFusionDB sugiere la integración ("Fusion") de diferentes modelos de datos y técnicas avanzadas de recuperación de información dentro de una única base de datos ("DB"). La idea es construir una plataforma versátil que unifique datos estructurados y no estructurados, incluyendo documentos textuales y objetos multimedia.

1. **Objetivo General**

Desarrollar un proyecto integral de base de datos que soporte tanto el modelo relacional basado en tablas como técnicas avanzadas de recuperación de información basadas en el contenido de documentos textuales y objetos multimedia.

1. **Objetivo especifico**

Comprender y aplicar de manera efectiva las técnicas de organización e indexación de archivos físicos con el fin de desarrollar un mini gestor de bases de datos, que optimice la gestión, almacenamiento y recuperación de datos estructurados dentro de un modelo relacional basado en tablas.

1. **Enunciado**

En grupos máximo de cinco integrantes, elegir un dominio de datos conformado por al menos dos archivos planos con datos reales (<https://www.kaggle.com/datasets>). Luego sobre ellos implementar tres técnicas de organización de archivos con sus principales operaciones: inserción, eliminación y búsqueda.

**Requerimientos generales**

* 1. Implementar tres de las siguientes técnicas de organización de archivos en memoria secundaria.
     1. Sequential File
     2. AVL File.
     3. ISAM-Sparse Index.
     4. Extendible Hashing.
     5. B+ Tree.
  2. Operaciones que se deben implementar:
     1. La búsqueda específica puede retornar más de un elemento que coincide con la key

Registro search(T key)

* + 1. La búsqueda por rango retorna todos los registros que se encuentran entre las dos llaves de búsqueda

vector<Registro> rangeSearch(T begin-key, T end-key)

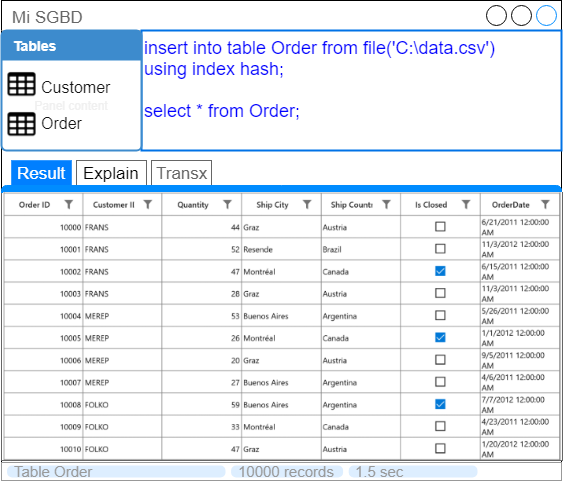
* + 1. Agregar un registro al archivo respetando la técnica de organización

bool add(Registro registro)

* + 1. Proponer un algoritmo de eliminación para cada técnica

bool remove(T key)

* 1. Implementar un parser de código SQL a sentencias ejecutables. Ejemplos de sentencias:
     + create table Customer from file “C:\data.csv” using index hash(“DNI”)
     + select \* from Customer where DNI = x
     + select \* from Customer where DNI between x and y
     + insert into Customer values (…)
     + delete from Customer where DNI = x
  2. Mostrar los resultados de forma amigable a usuario. Ver la siguiente GUI de referencia.



* 1. Opcionalmente puede obviar la GUI e implementar un bash sql (simulador de línea de comandos). Con nota máxima 18.

**Consideraciones de la implementación**

* 1. ***En el Sequential File, la función add(Registro registro)*** *d*ebe utilizar un espacio auxiliar para guardar los nuevos registros. Cuando el espacio auxiliar llegue a K registros, aplique un algoritmo de reconstrucción del archivo de datos manteniendo el orden físico de acuerdo a la llave seleccionada. Asegúrese de mantener los punteros actualizados.
  2. **En el ISAM** partir construyendo el índice estático hasta de 3 niveles a partir de un conjunto de datos. Para nuevas inserciones se generan los overflow pages (encadenamiento de páginas).
  3. **En el ISAM** de debe definir el factor de bloque tanto en las páginas de datos como en las páginas del índice.
  4. La función **rangeSearch** debe usar el índice para buscar el begin-key y luego recorrer los registros de acuerdo la estructura del índice. Esta función no es soportada por las técnicas de hashing.
  5. Use adecuadamente los conceptos de programación orientado a objetos y programación genérica para que el programa soporte cualquier dominio de datos.
  6. El backend se construye 100% en C++. Para el front-end se sugiere utilizar Python y alguna librería gráfica. Puede enlazar estos dos lenguajes con [Python Bindings](https://realpython.com/python-bindings-overview/).
  7. Es importante que todos participen en la implementación, se tomará en cuenta los commits en Github.

1. **Informe del proyecto**

* Archivo en Markdown / Wiki / Latex.
* Cuide la ortografía y consistencia en los párrafos.
* Al final del informe poner el **video explicando** el **funcionamiento** del programa, casos de uso, y aspectos importantes de la implementación. El video no debe exceder los 15 minutos y deben participar todos los integrantes del grupo.
* Aspectos de evaluación e informe:

|  |  |
| --- | --- |
| **Item** | **Descripción** |
| * Introducción   (2 pts) | * Objetivo del proyecto. * Descripción del dominio de datos a usar. * Resultados que se esperan obtener al aplicar las técnicas de indexación. |
| * Técnicas Utilizadas.   (9 pts) | * Describa brevemente las técnicas de indexación de archivos que ha elegido. * Explique el algoritmo de inserción, eliminación y búsqueda (use gráficos para un mayor entendimiento) * Se debe realizar un análisis comparativo teórico de las técnicas implementadas en base a los **accesos a memoria secundaria** tanto para las operaciones de inserción, búsqueda y eliminación. * En el código debe estar optimizado en el manejo de memoria secundaria. * Explicar claramente como se realizó el parser del SQL. |
| * Resultados Experimentales   (5 pts) | * Cuadro y/o gráfico comparativo de desempeño de las técnicas de indexación de archivos sobre el dominio de datos. **Tanto para inserción como para búsqueda**. * Considerar dos métricas: total de accesos a disco duro (read & write) y tiempo de ejecución en milisegundos. * Discusión y análisis de los resultados experimentales. |
| * Pruebas de uso y presentación   (4 pts) | * Presentar las pruebas de uso de la aplicación en interfaz gráfica amigable e intuitiva. * Recuerde mostrar la funcionalidad del aplicativo en el video. |

1. **Entregable**

* Los alumnos formarán grupos de máximo cinco integrantes.
* El código fuente del proyecto será alojado en GitHub.
* En el Canvas subir solo el **enlace público** del proyecto.
* La fecha límite de entrega es el --------.