**Proyecto TinySQLb Algoritmos y Estructura de Datos I**

**Nombre:** Jorge Alberto Marín Navarro

**Carné:** 2024174059

**Curso:** Algoritmos y Estructura de Datos I CE-1103

**Proyecto:** TinySQLb with C#

**Profesor:** Leonardo Araya

**Tabla de Contenidos**

1. **Introducción al desarrollo del proyecto ………………………………………………. 3**
2. **Breve Descripción del Problema ………………………………………………………3**
3. **Descripción de la Solución ..……………………………………………………………..3**3.1 Requerimientos e Implementación  
   3.2 Alternativas Consideradas  
   3.3 Limitaciones y Problemas Encontrados
4. **Diseño General…………………………………………………………………………….4**4.1 Diagrama de Clases UML  
   4.2 Patrones de Diseño Aplicados

Imagen que contiene nombre de la empresa

Descripción generada automáticamente

Documentación de TinySQLb

# **Introduction**

El presente documento describe el desarrollo de un sistema de base de datos denominado TinySQL, implementado para permitir la ejecución de consultas SQL básicas mediante el uso de PowerShell y C#. El proyecto está enfocado en la creación, manipulación, y administración de una base de datos sencilla que soporta operaciones SQL como CREATE, INSERT, DELETE, UPDATE y SELECT. TinySQL ha sido desarrollado para proporcionar una forma ligera y eficiente de manejar bases de datos sin la necesidad de un sistema de gestión de bases de datos convencional.

II. **Breve Descripción del Problema**

La gestión de datos estructurados es fundamental en cualquier aplicación moderna, sin embargo, muchas soluciones existentes requieren sistemas de gestión de bases de datos complejos que a menudo son difíciles de configurar y mantener. El objetivo de este proyecto es ofrecer una alternativa ligera para gestionar bases de datos, permitiendo la manipulación y consulta de datos de manera simple, todo a través de un entorno basado en scripts que pueda ser usado por cualquier usuario con conocimientos básicos de SQL.  
  
**III. Descripción de la Solución**

El sistema TinySQL permite realizar operaciones básicas sobre una base de datos, incluyendo creación de tablas, inserción de datos, actualización de registros, y eliminación de entradas, así como la consulta de información mediante el lenguaje SQL.

# **Requerimientos e implementación**

* Módulo de PowerShell para la Ejecución de Consultas (Execute-MyQuery): Se creó un módulo en PowerShell para permitir la ejecución de consultas SQL desde un script. Esta función admite parámetros para indicar el archivo de script, el puerto, y la dirección IP. Los resultados se muestran en la terminal en formato de tabla. Durante el desarrollo se utilizó QueryFile para gestionar los scripts SQL y procesar cada sentencia.
* Creación de Bases de Datos y Tablas: Para la creación de bases de datos se utilizaron comandos SQL estándar como CREATE DATABASE y CREATE TABLE. Estos comandos permiten gestionar la estructura de la base de datos y sus tablas de manera sencilla, generando archivos binarios que representan cada tabla.
* Índices en Columnas de Tablas: Se implementaron índices para columnas utilizando CREATE INDEX. Esto mejoró la eficiencia de las consultas, permitiendo evitar valores duplicados en las columnas indexadas y acelerando la búsqueda de registros. Alternativas consideradas incluyeron la implementación de un sistema de hashing para índices, pero finalmente se optó por árboles B debido a su eficiencia.
* Consultas SQL Básicas: Se desarrollaron operaciones de INSERT, UPDATE, DELETE, y SELECT. Los comandos se procesan utilizando C# y el almacenamiento se realiza en archivos binarios. Problemas encontrados incluyeron dificultades con el soporte de valores NULL y los tipos de datos DATE, que fueron resueltos implementando verificaciones manuales y utilizando representaciones estándar.
* Validaciones de Índice Único: Para evitar duplicados en campos únicos (como ID), se implementaron validaciones en la operación INSERT que aseguran que no se puedan agregar registros con valores repetidos.

**Alternativas Consideradas:**

* **Bases de Datos en Memoria vs Archivos Binarios:** Consideramos utilizar una base de datos en memoria para una mayor velocidad, pero decidimos que la persistencia de datos era más importante para nuestro caso de uso.
* **Patrones de Árboles para Índices:** Consideramos diferentes tipos de estructuras de árbol (BTree vs AVL), optando por BTree debido a sus mejores características de balanceo automático.

# **Limites y problemas encontrados**

* **Manejo de Tipos de Datos Complejos:** Una de las principales limitaciones fue la incapacidad de manejar tipos de datos complejos o relaciones entre tablas, limitándonos a tipos básicos como INTEGER, VARCHAR y DATETIME.
* **Consultas SQL Complejas:** El sistema solo soporta consultas básicas. No se implementó soporte para uniones (JOIN) o funciones agregadas (SUM, AVG, etc.).

##### Referencias

1. Elmasri, R., & Navathe, S. B. (2016). Fundamentals of database systems (7ª ed.). Pearson..

[2] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design patterns: Elements of reusable object-oriented software. Addison-Wesley..

[3] Richards, M., & Ford, N. (2020). Fundamentals of software architecture. O'Reilly Media.

[4] Snover, J. (2006). Monad manifesto. Microsoft. https://learn.microsoft.com/en-us/powershell/scripting/community/jeffrey-snover-the-monad-manifesto

[5] Microsoft. (n.d.). SQL documentation. https://learn.microsoft.com/en-us/sql/?view=sql-server-ver15

**Diagrama UML:**

Diagrama

Descripción generada automáticamente