Proyecto #2

Algoritmos y Estructura de Datos 1

Estudiantes:

Leandro José Ruíz Acuña
Carlos Andrés Jiménez Brenes

2023184994

2023125267

Instituto Tecnológico de Costa Rica

TinySQLDb

Profesor:

Leonardo Andrés Araya Martínez

## Resumen del Proyecto TinySQLDb

**Objetivo General:** Desarrollar un motor de bases de datos relacional sencillo para que los estudiantes comprendan su funcionamiento.

**Componentes del Proyecto:**

1. **Cliente:** Módulo de Powershell que permite ingresar consultas SQL y muestra resultados en formato tabla.
2. **Servidor:** Incluye:
    a. **API Interface:** Maneja la comunicación con el cliente en formato JSON.
    b. **Query Processing:** Valida la sintaxis y semántica de las consultas.
    c. **Stored Data Manager:** Accede a los archivos de datos de las tablas.

**Proceso de Ejecución:**

1. El cliente envía una consulta al API Interfaz.
2. El API Interfaz la procesa y la envía al Query Processor.
3. Este valida y coordina la ejecución con el Stored Data Manager, que accede a los datos.

El proyecto ofrece una experiencia práctica en el diseño e implementación de motores de bases de datos relacionales.

| Número de Requerimiento | Requerimiento | Solución |
|---|---|---|
| 000 | El cliente es un módulo de Powershell 7 que se carga en Powershell y que permite ingresar comandos para realizar operaciones sobre la base de datos. Se crea una función llamadaExecute-MyQuery con los siguientes parámetros: - QueryFile: Indica un path donde se encuentra el archivo con sentencias SQL por ejecutar. - Port: puerto en el que el API | Definir la Función: Crea una función que lea un archivo de sentencias SQL y se conecte al servidor utilizando sockets.<br><br>Leer el Archivo: La función debe leer el archivo especificado y separar las sentencias SQL utilizando el punto y coma como delimitador.<br><br>Conectar al Servidor: Utiliza las capacidades de |

| | | |
|---|---|---|
| | interface escucha - IP: dirección IP en el que el API interface escucha Por ejemplo: Execute-MyQuery –Query .\Script.tinysql -Port 8000 –IP 10.0.0.2 Los resultados se muestran en la terminal en formato de tabla. Powershell como tal, provee muchas facilidades para formatear objetos en formato tabla, puede investigar y usar dichas funciones. 15 Cada sentencia dentro del archivo se ejecuta una por una y se muestra el resultado de cada una. Por cada sentencia que se ejecuta se muestra el tiempo que se tardó el servidor en completarla. Cada sentencia del script se separa por punto y coma. | PowerShell para establecer una conexión con el servidor usando la dirección IP y el puerto proporcionados.<br>Ejecutar las Consultas: Envía cada consulta al servidor, mide el tiempo que toma cada ejecución y muestra el resultado en la terminal.<br><br>Mostrar Resultados: Usa las herramientas de PowerShell para formatear los resultados en una tabla y muestra el tiempo que tardó el servidor en procesar cada consulta. |
| 001 | Una base de datos se modela como una carpeta en el sistema de archivos. Archivos binarios dentro de esta carpeta corresponden a tablas de la base de datos. El system catalog es una carpeta con archivos binarios compartida para todas las bases de datos, que contiene metadata, es decir, datos sobre los datos. Almacena información como: 1. Listado de las bases de datos existentes (archivo SystemDatabases) 2. Tablas existentes en cada base de datos (archivo SystemTables) 3. | Estructura de Carpetas: Crea una carpeta principal que contenga todas las bases de datos. Dentro de cada base de datos, almacena archivos binarios que representen las tablas.<br><br>Sistema de Catalogación: Establece una carpeta específica para el System Catalog, donde guardarás archivos binarios que contengan metadata sobre las bases de datos, como: SystemDatabases: Lista de bases de datos. |

| | | |
|---|---|---|
| | Columnas de cada tabla (archivo SystemColumns) 4. Índices asociados a cada table (archivo SystemIndexes) Las tablas de System Catalog se pueden consultar con sentencias SELECT | SystemTables: Información sobre las tablas en cada base de datos.<br>SystemColumns: Detalles sobre las columnas en cada tabla.<br><br>SystemIndexes: Información sobre los índices de las tablas.<br>Consulta de Datos: Implementa funciones que permitan realizar consultas sobre estos archivos. Por ejemplo, las sentencias SELECT pueden traducirse en operaciones de lectura sobre los archivos del System Catalog.<br>Ejemplo de Consultas: Permite ejecutar consultas como SELECT * FROM SystemDatabases para obtener la lista de bases de datos, o SELECT * FROM SystemTables WHERE DatabaseName = 'miBaseDeDatos' para ver las tablas en una base de datos específica.<br><br>Manejo de Datos: Utiliza métodos de serialización para guardar datos en archivos binarios y deserialización para leerlos cuando se realicen las consultas. |
| 002 | Para crear una base de datos, la sentencia SQL es: CREATE DATABASE | Interpretar la Sentencia: La función encargada de manejar la creación de bases de datos debe reconocer la sentencia |

| | | |
|---|---|---|
| | | CREATE DATABASE <database-name>.<br>Extraer el Nombre de la Base de Datos: Se debe extraer el nombre de la base de datos a partir de la sentencia, asegurándose de que el nombre sea válido y no contenga caracteres no permitidos.<br>Verificar la Existencia: Antes de crear la nueva base de datos, verifica que no exista ya una base de datos con el mismo nombre para evitar conflictos.<br>Crear la Carpeta de la Base de Datos: Si el nombre es válido y no hay conflictos, crea una nueva carpeta en el sistema de archivos que llevará el nombre de la base de datos.<br>Actualizar el System Catalog: Después de crear la carpeta, actualiza el archivo SystemDatabases en el System Catalog para incluir la nueva base de datos. Esto podría implicar agregar una entrada en el archivo que registre el nombre de la base de datos y cualquier otra metadata relevante.<br>Devolver una Respuesta: Finalmente, devuelve un mensaje al usuario indicando que la base de datos se ha creado con éxito. |
| 003 | La sentencia: SET DATABASE establece el contexto en el cliente. El | Interpretar la Sentencia: La función que maneja las sentencias SQL debe |

| | | |
|---|---|---|
| | servidor únicamente valida que la base de datos solicitada exista. El cliente al recibir la respuesta del server establece el contexto para las siguientes sentencias SQL | reconocer SET DATABASE <database-name> y extraer el nombre de la base de datos.<br>Validar la Existencia: Envía una solicitud al servidor para verificar si la base de datos solicitada existe. Esto implica consultar el archivo SystemDatabases en el System Catalog para asegurarte de que el nombre indicado está registrado.<br>Recibir la Respuesta: El servidor debe responder indicando si la base de datos existe o no. Esta respuesta puede ser un mensaje simple de éxito o error.<br>Establecer el Contexto en el Cliente: Si la base de datos existe, el cliente debe actualizar su estado interno para recordar cuál es la base de datos activa. Esto permite que las siguientes sentencias SQL se ejecuten en el contexto de la base de datos seleccionada.<br>Confirmar al Usuario: Informa al usuario sobre el cambio de contexto, indicando que ahora está trabajando con la base de datos especificada.<br>Ejecutar Consultas: Para las sentencias SQL que se ejecuten posteriormente, asegúrate de que utilicen el contexto de la base de datos activa. |
| 004 | Para crear una tabla para en una base de datos | Interpretar la Sentencia: La función que maneja las |

| | | |
|---|---|---|
| | específica, se usa la sentencia CREATE TABLE AS ( column-definition ); column-definition: type nullability constraint type: INTEGER \| DOUBLE \| VARCHAR(length) \| DATETIME Integer y Double corresponde a los enteros y double tradicionales. Varchar es un string pero debe ser de tamaño fijo, por lo que se deberá especificar el tamaño máximo al crear la tabla Datetime es un formato fecha y hora. | sentencias SQL debe reconocer CREATE TABLE <table-name> AS (column-definition) y extraer tanto el nombre de la tabla como las definiciones de las columnas.<br>Validar el Contexto de la Base de Datos: Asegúrate de que se haya seleccionado una base de datos activa mediante la sentencia SET DATABASE. Si no hay una base de datos activa, devuelve un error.<br>Definir las Columnas: Extrae y valida cada definición de columna, que debe incluir:<br>Nombre de la Columna: Debe ser único dentro de la tabla.<br>Tipo de Dato: Debe ser uno de los tipos permitidos (INTEGER, DOUBLE, VARCHAR(length), DATETIME).<br>Nullabilidad: Determina si la columna puede aceptar valores nulos.<br>Restricciones: Verifica si hay restricciones adicionales que deben aplicarse, como claves primarias.<br>Crear la Estructura de la Tabla:<br>Crea un archivo binario que represente la tabla dentro de la carpeta de la base de datos seleccionada. Este archivo almacenará la metadata de la tabla, incluyendo la definición de columnas. |

| | | |
|---|---|---|
| | | Actualizar el System Catalog:<br>Agrega una entrada en el archivo SystemTables del System Catalog que registre la nueva tabla y sus columnas. Esto permite que el sistema tenga un seguimiento de las tablas disponibles en cada base de datos.<br>Devolver una Respuesta:<br>Informa al usuario que la tabla ha sido creada con éxito. |
| 005 | DROP TABLE 5 Elimina una tabla de la base de datos establecida en el contexto, siempre y cuando la tabla esté vacía | interpretar la Sentencia:<br>La función encargada de manejar las sentencias SQL debe reconocer DROP TABLE <table-name> y extraer el nombre de la tabla a eliminar.<br>Verificar el Contexto de la Base de Datos: Asegúrate de que se haya seleccionado una base de datos activa. Si no hay una base de datos establecida, devuelve un error.<br>Comprobar la Existencia de la Tabla: Verifica si la tabla especificada existe en la base de datos actual. Esto implica consultar el archivo correspondiente en el sistema.<br>Validar que la Tabla esté Vacía: Antes de eliminar la tabla, comprueba que no contenga ningún dato. Si la tabla tiene registros, devuelve un error indicando que no se puede eliminar porque no está vacía. |

| | | |
|---|---|---|
| | | Eliminar la Tabla: Si la tabla existe y está vacía, procede a eliminar el archivo que representa la tabla de la base de datos y, si es necesario, elimina su entrada del catalogo del sistema.<br>Confirmar al Usuario: Informa al usuario que la tabla ha sido eliminada exitosamente. |
| | | |
| 006 | CREATE INDEX ON () OF TYPE Type: BTREE, BST<br>Permite crear índices sobre una columna de la tabla. Para poder agregar a un índice a una tabla con datos, se debe verificar que no haya valores repetidos. Posteriormente, no se permite ingresar valores repetidos para dicha columna. El tipo BTREE genera un árbol B en memoria utilizando los datos de dicha columna. El tipo BST, genera un árbol binario de búsqueda con los datos de dicha columna. Solo se puede un índice a la vez en una sola columna de la tabla. El propósito del índice es que se tenga una estructura más liviana en memoria que permita hacer búsquedas rápidas y que indique donde está el registro completo en el archivo. Cada vez que se crea un índice, se registra en el system catalog. Los índices ya existentes, se crean en memoria cuando | Interpretar la Sentencia: La función que maneja las sentencias SQL debe reconocer CREATE INDEX <index-name> ON <table-name>(<column-name>) OF TYPE <type> y extraer los nombres del índice, la tabla y la columna, así como el tipo de índice.<br>Verificar el Contexto de la Base de Datos: Asegúrate de que hay una base de datos activa seleccionada. Si no hay una base de datos establecida, devuelve un error.<br>Validar la Existencia de la Tabla: Comprueba que la tabla especificada existe en la base de datos. Si no existe, devuelve un error.<br>Comprobar Valores Repetidos: Antes de crear el índice, verifica que no haya valores duplicados en la columna seleccionada. Si hay duplicados, devuelve un error indicando que no se puede crear el índice.<br>Crear el Índice: |

| | | |
|---|---|---|
| | el servidor inicia. Es decir, si en el system catalog se indica que existe el índice x, si el servidor se reinicia, se debe volver a crear en memoria. | Dependiendo del tipo especificado (BTREE o BST), genera la estructura de datos correspondiente en memoria utilizando los valores de la columna.<br>Un índice BTREE crea un árbol B, mientras que un índice BST crea un árbol binario de búsqueda.<br>Registrar el Índice en el System Catalog: Añade la información del nuevo índice al archivo correspondiente en el System Catalog, registrando el nombre del índice, la tabla, la columna y el tipo.<br>Manejo de Reinicios del Servidor: Implementa la lógica para que, al iniciar el servidor, se lean los índices del System Catalog y se reconstruyan en memoria si ya existen.<br>Confirmar al Usuario: Informa al usuario que el índice ha sido creado exitosamente. |
| 007 | Para seleccionar filas de una tabla específica: SELECT * \| FROM [WHERE where-statement] [ORDER BY ] where-statement: compare-operator [] compare-operator: >, <, =, like, not El resultado del SELECT se muestra como una tabla en la interfaz del cliente. El query processor si detecta que hay una columna especificada en el WHERE para la cual hay un índice, se utiliza el índice para buscar | Interpretar la Sentencia: La función encargada de manejar las sentencias SQL debe reconocer SELECT * \| <columns> FROM <table-name> [WHERE where-statement] [ORDER BY <column> <asc \| desc>] y extraer los elementos necesarios, como las columnas a seleccionar, el nombre de la tabla y cualquier cláusula WHERE o ORDER BY. |

| | | |
|---|---|---|
| | primero. Si no hay índice, se hace una búsqueda secuencial. Si se especifca el Order by, se utiliza Quicksort para ordenar el resultado de la consulta. Puede ser ascendente o descendente | Verificar el Contexto de la Base de Datos: Asegúrate de que hay una base de datos activa seleccionada. Si no hay una base de datos establecida, devuelve un error.<br>Validar la Existencia de la Tabla: Comprueba que la tabla especificada existe en la base de datos. Si no existe, devuelve un error.<br>Preparar la Consulta:<br>Si se especifican columnas, selecciona solo esas; si se usa *, selecciona todas las columnas.<br>Si se incluye una cláusula WHERE, verifica la sintaxis y asegúrate de que la columna mencionada sea válida.<br>Utilizar Índices (si están disponibles):<br>Si hay un índice para la columna mencionada en la cláusula WHERE, utiliza el índice para realizar una búsqueda más eficiente.<br>Si no hay un índice, realiza una búsqueda secuencial a través de las filas de la tabla.<br>Filtrar Resultados: Aplica la cláusula WHERE para filtrar los resultados según los criterios especificados.<br>Ordenar Resultados (si se especifica):<br>Si se incluye la cláusula ORDER BY, utiliza el algoritmo Quicksort para ordenar los resultados. Permite ordenar en orden ascendente o |

| | | |
|---|---|---|
| | | descendente según lo indicado.<br>Mostrar Resultados: Presenta los resultados de la consulta en formato de tabla en la interfaz del cliente.<br>Manejo de Errores: Incluye manejo de errores para situaciones como consultas inválidas o problemas al acceder a los datos. |
| 008 | UPDATE <table-name> SET <column-name> = <new-value> [WHERE where-statement]<br>Actualiza las filas de la tabla que cumplan la condición del WHERE. En caso que WHERE no se especifique, se actualizan todas las filas. 10 La actualización se realiza sobre las columnas indicadas en el SET. El query processor si detecta que hay una columna especificada en el WHERE para la cual hay un índice, se utiliza el índice para buscar primero. Si no hay índice, se hace una búsqueda secuencial. Si el SET actualiza alguna columna indizada, el índice asociado deberá actualizarse. | Pasos para la Implementación de UPDATE<br>Interpretar la Sentencia: La función debe reconocer y descomponer la sentencia UPDATE <table-name> SET <column-name> = <new-value> [WHERE where-statement], extrayendo el nombre de la tabla, la columna a actualizar, el nuevo valor y cualquier cláusula WHERE.<br>Verificar el Contexto de la Base de Datos: Asegúrate de que hay una base de datos activa seleccionada. Si no hay ninguna, devuelve un error.<br>Validar la Existencia de la Tabla: Comprueba que la tabla especificada existe en la base de datos. Si no existe, devuelve un error.<br>Validar la Columna: Asegúrate de que la columna que se desea actualizar existe en la tabla. Si no, informa al |

| | | |
|---|---|---|
| | | usuario con un mensaje de error.<br>Preparar la Consulta:<br>Si se incluye una cláusula WHERE, verifica la sintaxis y asegúrate de que la columna mencionada sea válida. Si no hay cláusula WHERE, todas las filas de la tabla serán afectadas.<br>Uso de Índices (si están disponibles):<br>Si hay un índice para la columna especificada en la cláusula WHERE, utilízalo para realizar una búsqueda eficiente de las filas que cumplen con la condición.<br>Si no hay índice, realiza una búsqueda secuencial a través de las filas de la tabla.<br>Actualizar Filas:<br>Aplica la actualización al nuevo valor en las filas que cumplan con la condición del WHERE. Si no hay cláusula WHERE, actualiza todas las filas.<br>Actualizar Índices (si corresponde):<br>Si la columna que se está actualizando es una columna indizada, actualiza el índice asociado para reflejar los cambios.<br>Confirmar al Usuario:<br>Informa al usuario sobre el número de filas que han sido actualizadas.<br>Manejo de Errores:<br>Implementa manejo de errores para situaciones como intentos de actualización de columnas |

| | | |
|---|---|---|
| | | no válidas o problemas al acceder a los datos. |
| 009 | DELETE FROM [WHERE where-statement] Elimina las filas de la tabla que cumplan la condición del WHERE. En caso que WHERE no se especifique, se eliminan todas las filas. El query processor si detecta que hay una columna especificada en el WHERE para la cual hay un índice, se utiliza el índice para buscar primero. Si no hay índice, se hace una búsqueda secuencial. Al eliminar filas, si hay índices asociados, deben actualizarse | Interpretar la Sentencia:<br>La función debe reconocer y descomponer la sentencia DELETE FROM <table-name> [WHERE where-statement], extrayendo el nombre de la tabla y cualquier cláusula WHERE.<br>Verificar el Contexto de la Base de Datos:<br>Asegúrate de que hay una base de datos activa seleccionada. Si no hay ninguna, devuelve un error.<br>Validar la Existencia de la Tabla:<br>Comprueba que la tabla especificada existe en la base de datos. Si no existe, devuelve un error.<br>Preparar la Consulta:<br>Si se incluye una cláusula WHERE, verifica la sintaxis y asegúrate de que la columna mencionada sea válida. Si no hay cláusula WHERE, todas las filas de la tabla serán eliminadas.<br>Uso de Índices (si están disponibles):<br>Si hay un índice para la columna especificada en la cláusula WHERE, utilízalo para realizar una búsqueda eficiente de las filas que cumplen con la condición.<br>Si no hay índice, realiza una búsqueda secuencial a través de las filas de la tabla.<br>Eliminar Filas: |

| | | |
|---|---|---|
| | | Elimina las filas que cumplen con la condición del WHERE. Si no hay cláusula WHERE, elimina todas las filas de la tabla.<br>Actualizar Índices (si corresponde):<br>Si hay índices asociados a las columnas eliminadas, actualiza esos índices para reflejar los cambios, eliminando las referencias a las filas que han sido borradas.<br>Confirmar al Usuario:<br>Informa al usuario sobre el número de filas que han sido eliminadas.<br>Manejo de Errores:<br>Implementa manejo de errores para situaciones como intentos de eliminación de filas no válidas o problemas al acceder a los datos. |
| 010 | NSERT INTO VALUES(,...)<br>Inserta en la tabla especificada los valores para cada columna en el orden de creación de la tabla. Valida que los tipos de dato especificados sean los correctos y actualiza los índices asociados. Las columnas DateTime se especifican en String pero internamente se parsean a DateTime | Interpretar la Sentencia:<br>La función debe reconocer y descomponer la sentencia INSERT INTO <table-name> VALUES (<values>, <values>, ...), extrayendo el nombre de la tabla y los valores a insertar.<br>Verificar el Contexto de la Base de Datos:<br>Asegúrate de que hay una base de datos activa seleccionada. Si no hay ninguna, devuelve un error.<br>Validar la Existencia de la Tabla:<br>Comprueba que la tabla especificada existe en la |

| | | |
|---|---|---|
| | | base de datos. Si no existe, devuelve un error.<br>Validar los Valores:<br>Verifica que la cantidad de valores proporcionados coincida con el número de columnas de la tabla.<br>Asegúrate de que los tipos de datos de los valores coincidan con los tipos de las columnas:<br>Si hay columnas de tipo DateTime, parsea los strings a DateTime internamente.<br>Insertar Valores:<br>Añade los valores a la tabla en el orden de creación de las columnas.<br>Si hay restricciones de integridad (como claves primarias o foráneas), asegúrate de que los nuevos datos las respeten.<br>Actualizar Índices:<br>Si la tabla tiene índices asociados, actualiza esos índices para incluir la nueva fila.<br>Confirmar al Usuario:<br>Informa al usuario que la inserción fue exitosa y, si es posible, indica el ID o la clave de la nueva fila (si aplica).<br>Manejo de Errores:<br>Implementa manejo de errores para situaciones como inserciones inválidas o problemas al acceder a los datos. |

Diagrama UML