Compiladores
Laboratorio de Análisis Léxico- Lex

Generalidades

Objetivo: Aplicar el proceso de análisis léxico para identificar los tokens del lenguaje SQL y los errores léxicos.
Herramienta: Lex o Flex y C o C++ bajo el sistema operativo Linux Ubuntu.
Entrada: Archivo de texto con instrucciones SQL llamado **entrada.txt.**
Salida: Archivo de texto llamado **salida.txt** con instrucciones SQL donde se identifique cada token y se listen los identificadores y el número de error encontrados
Manual: Se debe incluir un **manual** de instrucciones en PDF donde describa cada uno de los pasos de compilación y ejecución.
Fecha de inicio: Septiembre 7 de 2023.
Fecha de entrega: Octubre 6 de 2023.
Nombre programa: LAB01 (Fuente Lex).
LAB01 (Fuente C).
LAB01 (Ejecutable)
Integrantes: 3 ESTUDIANTES POR GRUPO (Sustentan por lo menos uno de los integrantes del grupo, si es necesario).
Los archivos se deben entregar comprimidos en un archivo llamado **LAB01_Nombre1Apellido1_Nombre2Apellido2_Nombre3Apellido3.zip. No se acepta en otro formato de compresión.** El manual debe ser incluido en este archivo comprimido.

Ejecución: El archivo de **entrada.txt** debe ser dado como un parámetro a continuación del nombre del programa ejecutable. La salida del programa debe ser direccionada (>) a un archivo de salida llamado **salida.txt**. Cada línea del archivo de entrada contiene sentencias SQL que pueden estar relacionadas entre sí. El archivo de entrada **NO** se debe pedir en forma interactiva.
Ejemplo de ejecución:
.\LAB01 entrada.txt > salida.txt
Donde **LAB01** es el programa ejecutable.

Descripción de la gramática

El analizador léxico recibirá un archivo de texto con instrucciones SQL que puede contener los siguientes tokens:

- ✓ Comandos: Dentro de las cuales encontramos los comandos de creación y eliminación de tablas (CREATE TABLE, DROP TABLE), de búsqueda de información (SELECT, WHERE, GROUP BY, ORDER BY), de manipulación de información (INSERT, DELETE, UPDATE) , funciones (MAX, MIN,AVG,COUNT) y palabras reservadas para los comandos (INTO, VALUES, FROM,SET, ASC, DESC)

- ✓ Identificadores: En donde se encuentran los nombres de las tablas y columnas. Los nombres pueden asociarse a las variables que son cualquier cadena que inicie con una letra y que puede continuar con una letra o un número.

- ✓ Números: Pueden ser de tipo entero o decimal.

- ✓ Cadenas: Secuencia de caracteres entre comillas, no son variables o identificadores.
- ✓ Tipos de Datos: En donde se encuentran valores de tipo INTEGER, DECIMAL, VARCHAR.
- ✓ Operadores: Donde se encuentran los operadores aritméticos, de condición y lógicos: suma (+), resta (-), multiplicación (*), división (/), igualdad (==), diferencia (<>), mayor que (>), menor que (<), mayor o igual que (>=), menor o igual que (<=), conjunción (AND) y disyunción (OR).
- ✓ Separadores: Paréntesis que abre (*(*), paréntesis que cierra(*)*), coma (,), punto y coma (;), asignación (=).
- ✓ Caracteres especiales: asterisco (*).

Estructuras

Se especifica a continuación la estructura de las instrucciones que se evaluarán teniendo en cuenta que SQL tiene muchas funciones y comandos. Lo que se encuentra en *[CURSIVA]* implica un componente opcional, lo que se encuentra en **negrilla** implica una palabra reservada y el símbolo | indica las diferentes opciones que puede llevar una instrucción.

Creación o eliminación de tablas

- ✓ **CREATE table** identificador(identificador **TipoDato**(número)*[, identificador* ***TipoDato****(número), …]*);
- ✓ **DROP table** identificador;

Inserción, Eliminación, Actualización

- ✓ **INSERT INTO identificador[identificador …] VALUES(identificador [, identificador …])**
- ✓ **DELETE FROM identificador WHERE condición [AND condición, OR condición, …]];**
- ✓ **UPDATE identificador SET identificador=identificador|número WHERE condición [AND condición, OR condición, …]];**

Búsqueda básica, con funciones y combinada

- ✓ **SELECT *|identificador[,identificador,…] FROM identificador ;**
- ✓ **SELECT Función(identificador) FROM identificador ;**
- ✓ **SELECT Función(identificador)|identificador[,Función(identificador)|identificador,…]] FROM identificador ;**

Búsquedas Condicionadas, agrupadas y ordenadas

- ✓ **SELECT *|identificador[,identificador,…] FROM identificador WHERE condición [AND condición, OR condición, …];**
- ✓ **SELECT *|identificador[,identificador,…] FROM identificador GROUP BY identificador;**
- ✓ **SELECT *|identificador[,identificador,…] FROM identificador ORDER BY identificador[,identificador, …]] ASC|DESC;**

- ✓ **SELECT *|identificador[,identificador,…] FROM identificador [WHERE condición [AND condición, OR condición, …]][GROUP BY identificador] [ORDER BY identificador[,identificador, …]] ASC|DESC];**

Descripción del archivo de salida

La salida debe mostrar las instrucciones del archivo de entrada identificando el tipo de token encontrado, a excepción de las palabras reservadas. A continuación, se detallan las especificaciones:

- ✓ Cada una de las palabras reservadas y comandos de las instrucciones debe mostrarse en mayúscula.
- ✓ Los identificadores deben ir numerados. En caso de encontrar uno repetido debe tener la misma numeración en cada aparición. Al terminar el análisis léxico se deben mostrar cuántos y cuáles fueron los identificadores encontrados.
- ✓ Para los números, operadores y separadores se debe especificar el tipo de token encontrado.
- ✓ Para los números identificar si son enteros o decimales

Errores Léxicos

- ✓ Un token no especificado en la gramática debe indicar error en el lugar donde fue encontrado.
- ✓ Un error en la estructura de la instrucción no es un error léxico
- ✓ Si hay error léxico en algún punto del archivo, se debe continuar el análisis hasta el final.
- ✓ Al terminar el análisis se debe mostrar cuántos errores fueron encontrados.

Ejemplos

Entrada.txt

CREATE Table EMPLEADO(nombre VARCHAR(20), cédula INTEGER, salario

DECIMAL(2));

INSERT INTO Empleado VALUES('Juan',1045268154, 12aab);

SELECT FROM a;

SELECT COUNT() FROM tabla;

SELECT nombre, MAX(salario) FROM Empleado;

UPDATE SET nombre='Carlos' FROM Empleado WHERE cédula=12.5;

SELECT * FROM Empleado WHERE salario<2000000 ORDER BY nombre ASC . ;

Salida.txt

CREATE TABLE id1=Empleado **parabre=**(**id2=**nombre **VARCHAR parabre=**(**entero=**20 **parcierr=**) **coma=**, **id3=**cédula **INTEGER coma=**, **id4=**salario **DECIMAL parabre=**(**entero=**2 **parcierr=**) **parcierr=**) **puntcoma=**;

INSERT INTO id1=Empleado **VALUES parabre=**(**cadena=**‘Juan’ **coma=** , **entero=** 1045268154 **coma=**, **ERROR=** 12aab **parcierr=**) **puntcoma=**; **SELECT**

FROM id5=a **puntcoma=**;

SELECT COUNT parabre=(**parcierr=**) **FROM id6=**tabla **puntcoma=**;

SELECT id2= nombre **coma=** , **MAX parabre=**(**id4=**salario **parcierr=**) **FROM id1**=Empleado **puntcoma=**;

UPDATE SET id2=nombre **asign=** = **cadena=** ’Carlos’ **FROM id1=**Empleado **WHERE id3**=cédula **asign=** = **decimal=** 12.5 **puntcoma=**;

SELECT asterisco=* **FROM id1=**Empleado **WHERE id4=**salario **menorq=**< **entero=**2000000 **ORDER BY id2=** nombre **ASC ERROR= . puntcoma=**;

6 Identificadores

Id1=Empleado
Id2=nombre
Id3=cédula
Id4=salario
Id5=a
Id6=tabla

2 Errores léxicos