UNIVERSIDAD COOPERATIVA DE COLOMBIA

INGENIERIA DE SOFTWARE

Actividad Final Integrada (Momento 3)

Diseño de un DSL Dinámico para Consultas sobre Archivos CSV (con ANTLR4 y Python)

Objetivo General

Diseñar, implementar e interpretar un **mini-compilador** funcional basado en un **Lenguaje de Dominio Específico (DSL)** usando **ANTLR4** y **Python**, que permita realizar **consultas dinámicas** sobre un archivo **CSV** de datos estructurados, mediante instrucciones secuenciales tipo comando.

Tema Asignado

Tema	Descripción	Campos CSV
Administración de Eventos Culturales	Organización de eventos culturales y artísticos.	id_evento, nombre_evento, tipo_evento, fecha, lugar, organizador, costo_entrada, cantidad_asistentes, patrocinadores, estado_evento

Descripción de la Actividad

- Diseñar su propio DSL basado en comandos secuenciales.
- Implementar un compilador que permita acumular instrucciones de filtrado y procesarlas de manera diferida.
- Procesar un archivo CSV de datos estructurados, permitiendo realizar consultas dinámicas mediante diferentes scripts.

Ejemplo de un script:

El DSL debe soportar instrucciones como:

```
load "empleados.csv";
filter column "edad" > 25;
aggregate count column "id";
aggregate average column "salario";
aggregate sum column "dias_laborados";
print;
```

Con la instrucción anterior el script debería:
- Cargar el archivo `empleados.csv`.

- Filtra empleados mayores de 25 años.

- Cuenta cuántos empleados cumplen la condición.

- Calcula el salario promedio de esos empleados.

- Suma el total de días laborados de esos empleados.

- Imprime todos los resultados de agregaciones al final.

Importante:

- Los filtros y operaciones de agregación no deben ejecutarse inmediatamente.
- Todas las operaciones deben acumularse y ejecutarse solo cuando se reciba print;.

Requisitos del Proyecto

Requisito	Detalle
Formato de Datos	Archivo **CSV** obligatorio.
Estructura del CSV	10 campos (columnas) relevantes según el tema asignado.
Cantidad de Datos	300 registros sintéticos [Los datos pueden ser generados de forma automática o manual]
Instrucciones mínimas en el DSL	load, filter, aggregate, print. *Aggregate(COUNT, SUM, AVERAGE)
Consultas a realizar	Mínimo 40 scripts distintos, usando combinaciones de instrucciones.
Combinación de filtros	Se deben poder aplicar uno o más filtros en un mismo script.
Lectura dinámica	Todos los scripts deben ejecutarse sin modificar la gramática ni el compilador.
Readme.txt	Documentación de los procesos a ejecutar

Operaciones que debe soportar el DSL

- Cargar datos desde un archivo CSV (load).
- Aplicar filtros sobre campos con operadores (filter column "campo" operador valor).

  OPERATOR: '>=' | '<=' | '>' | '<' | '==' | '!=' ;

  OPERADORES COMBINADOS ENTRE FILTROS: AND, OR

- Realizar **operaciones de agregación**:

  COUNT, SUM, AVERAGE, BETWEEN

- Imprimir resultados (print).

  Verificar proceso de extensión de funcionalidades:
  https://asigcompiladores.vercel.app/modules/Actividad_final_5.html

Aclaraciones importantes

- El DSL debe estar basado en instrucciones secuenciales, tipo comando a comando.
- Los filtros deben acumularse: no se aplican en el momento de su declaración, sino todos juntos al momento del print.
- El procesamiento de datos solo debe realizarse en el comando print;.
- La gramática y el compilador deben ser fijos: no pueden cambiarse entre consultas.
- Cada uno de los 40 scripts deberá mostrar distintas combinaciones de filtros, agregaciones y campos, sin modificar el diseño del DSL.
- Se debe demostrar mediante los scripts que el lenguaje es flexible y dinámico, sin necesidad de modificar su definición base.

Obligaciones adicionales

Cada equipo debe presentar:

Requisito	Descripción
Lexer generado	Definido en ANTLR4 (.g4) para reconocer los tokens del lenguaje.
Parser generado	Definido en ANTLR4 (.g4) para analizar las instrucciones según las reglas sintácticas.
Parse Tree	Mostrar al menos un **Parse Tree** real de un script de prueba como evidencia del análisis sintáctico.

Sustentación del Proyecto

- Cada bina deberá realizar una **sustentación obligatoria** de su proyecto
- Durante la sustentación, deberán:
  - Explicar el diseño de su DSL y su gramática.
  - Ejecutar varios scripts de prueba en tiempo real, demostrando:
    - La carga correcta del CSV.

- La acumulación dinámica de filtros.
- La correcta ejecución al aplicar print.

  - Responder preguntas relacionadas con su Lexer, Parser y Parse Tree.
- La sustentación tiene un valor fundamental dentro de la evaluación final.

Entregables

No.	Entregable	Descripción
1	Documento de definición del DSL	Diseño formal, explicación de instrucciones y gramática .g4.
2	Código fuente	Lexer, parser e intérprete en Python.
3	Archivo CSV	Dataset propio con 10 columnas y 300 registros.
4	40 scripts DSL	Scripts distintos combinando filtros, agregaciones y resultados.
5	Parse Tree	Captura o exportación de un árbol de análisis sintáctico de prueba.
6	Informe técnico	Documentación completa del proceso de diseño, implementación, pruebas y resultados.
7	Sustentación oral	Demostración y defensa del proyecto en la fecha asignada.

- Los entregables del 1 al 6 se deben condensar en el subdirectorio /doc, del directorio donde se realiza el proyecto; este sé publicará en el Campus Virtual, para su revisión.
- Para cumplir con el numeral 5, es necesario configurar su Antlr4 en local y lanzar la visualización -gui.
  Guiare con la siguiente url:
  https://asigcompiladores.vercel.app/modules/semana3d.html

Criterios de Evaluación

Criterio	Ponderación
Definición y formalización del DSL	15%
Implementación correcta de Lexer y Parser (ANTLR4)	20%
Generación y evidencia de Parse Tree	15%
Ejecución dinámica de consultas sobre CSV	20%
Correcta acumulación y ejecución de filtros y agregaciones	15%

Criterio	Ponderación
Documentación y presentación del informe final	5%
Sustentación del proyecto	10%

Recursos Recomendados

- Libro: *The Definitive ANTLR4 Reference* – Terence Parr.
- Documentación oficial ANTLR4: ANTLR4 Documentation.
- Ejemplos prácticos vistos en clase sobre Lexer, Parser, Listener y Visitor.

Mg. Gustavo Sánchez Rodriguez
Docente Universidad Cooperativa de Colombia