INF-253 Lenguajes de Programación
Tarea 1: Python

Profesor: Roberto Nicolas Diaz Urra
Ayudante Cátedras: Sebastián Godínez
Ayudante Tareas: Monserrat Figueroa - Sebastián Campos

Lunes 19 de Agosto de 2019

1. Historia

Python, C, Java , Scheme y Prolog. Hace muchos años los cinco lenguajes convivían en armonía, pero todo cambió cuando la nación del P = NP atacó. Sólo JLML, maestro de los 5 lenguajes podía resolver este problema, pero cuando el mundo más lo necesitaba, se echó mate. Pero aún hay esperanza, tú como alumno en tu humilde posición no puedes igualar el poder del antiguo salvador del mundo, pero cuenta la leyenda que existen dispersas por el universo cinco gemas conocidas como: "Las gemas de LP del infinito". Si alguien lograra conseguirlas todas podría ser capaz de equiparar el poder del antiguo salvador, y quizás poder evitar una catástrofe de proporciones bíblicas

Tú, lleno de determinación, te alistas para comenzar tu aventura, por tus conocimientos previos, sabes que la primera gema a conseguir es la gema de las E X P R E S I O N E S R E G U L A R E S, la cual se encuentra en el distrito Pythonia.

2. Objetivos

El alumno aplicará conceptos de **Expresiones regulares** en la implementación de un programa el cual procesará comandos SQL sobre archivos CSV[1], al analizar sintácticamente diferentes consultas para posteriormente obtener resultados dependiendo de estas.

3. Requerimientos

- El alumno debe utilizar **Python 3** para el desarrollo de esta tarea.
- El programa debe utilizar expresiones regulares para el análisis del código. Pueden consultar la documentación del módulo **re** en la siguiente URL: `https://docs.python.org/3/library/re.html`
- El programa recibirá uno o más archivos .csv con diversos nombres.
- El programa debe imprimir o modificar el contenido de los archivos .csv según se pida.

[1] `https://es.wikipedia.org/wiki/Valores_separados_por_comas`

4. Funcionalidad del Programa

Utilizando sus conocimientos de expresiones regulares y procesamiento de archivos, el programa deberá ser capaz de procesar y aplicar **múltiples** comandos SQL sobre uno o varios archivos .csv. Estos comandos serán ingresados por consola. Además, el programa debe ser capaz de reconocer si un comando no presenta la sintaxis adecuada, o si el archivo/columnas consultados existen.

5. Introducción a lenguaje SQL

SQL o Structured Query Language es un lenguaje de dominio específico utilizado en programación, diseñado para administrar y recuperar información de sistemas de gestión de bases de datos relacionales. En otras palabras, es el lenguaje utilizado para la obtención, actualización, inserción, borrado, y otras operaciones de datos en una base de datos de tipo relacional.

Para motivos de ésta tarea, sólo se usarán las declaraciones SELECT, INSERT y UPDATE, las cuales se explicarán a continuación.

5.1. SELECT

```
"SELECT" (columna {"," columna} | "*") "FROM" tabla
["INNER JOIN" tabla]
["WHERE" (columna "=" valor | columna "=" columna) {("AND"|"OR") (columna "=" valor | columna "=" columna)}]
["ORDER BY" columna ("ASC"|"DESC")]
```

El comando SELECT, se utiliza para obtener información de una tabla de datos que siga un formato específico. SELECT empieza señalando que columnas se desean consultar, se puede pedir una o varias columnas dependiendo de lo que se necesite o se utiliza el símbolo * para señalar que se desean consultar todas las columnas. Luego, se elige de que tabla se desean obtener estas columnas. Posteriormente, se puede especificar si los datos extraídos tengan un valor específico (o varios) en alguna de sus columnas, estos valores pueden ser números enteros o flotantes, como también strings encerrados en comillas simples . Finalmente, se pueden ordenar los datos mostrados según el orden ascendente o descendente de una columna especificada (ORDER BY).

El AND posee precedencia con respecto al OR.

En el caso de usar INNER JOIN, se obtienen los resultados de la unión de dos tablas diferentes, para esto se tiene que especificar en el WHERE una condición con las columnas que se igualarán en la unión.

5.2. INSERT

```
"INSERT INTO" tabla "("columna {"," columna2 }")"
"VALUES ("valor {"," valor2 }");"
```

El comando INSERT, se utiliza para insertar datos nuevos a una tabla. Su estructura es bastante simple, se escribe entre paréntesis el nombre de las columnas a las cuales se les asignará un valor, luego se coloca la keyword VALUES para, finalmente, escribir los valores que se insertarán también entre paréntesis.

Al especificar menos columnas que las existentes en la tabla, las columnas faltantes tomarán como valor por defecto un string vacío. Sin embargo, si existe un número de columnas distinto

al de valores a insertar se considerará como un error de sintaxis.

5.3. UPDATE

```
"UPDATE" tabla
"SET" columna "=" valor {"," columna "=" valor}
"WHERE" columna "=" valor {("AND" columna "=" valor |"OR" columna"="valor)}";"
```

El comando UPDATE, sirve para actualizar datos existentes en una tabla, retornando el número de filas actualizadas. Su estructura comienza especificando la tabla en la cual se actualizarán los datos, luego se procede a señalar los datos a actualizar con sus respectivos nuevos valores para, finalmente, señalar una serie de condiciones que deben cumplirse para actualizar los datos en la tabla.

Nota: Las sentencias anteriores son en una línea. Por cuestiones de espacio fueron escritas en múltiples líneas

6. Ejemplo

A continuación se mostrarán algunos ejemplos de comandos SQL aplicados a las siguientes tablas:

Nombre	Rol	Ramo	Sigla	Nota
Clemente Aguilar	201773580-3	Matemáticas 1	Mat-021	71
Ignacio Aedo	201773531-4	Lenguajes de Programación	INF-253	54
Gonzalo Fernandez	201673557-4	Lenguajes de Programación	INF-253	72
Gabriel Carmona	201773509-0	Base de datos	INF-239	78
Gabriel Carmona	201773509-0	Sistemas Operativos	INF-246	70

Notas

Nombre	Rut	Rol	Teléfono	Edad
Clemente Aguilar	20207131-k	201773580-3	988714567	20
Ignacio Aedo	19312897-4	201773531-4	912345678	20
Gonzalo Fernandez	19992851-1	201673557-4	987654321	21
Gabriel Carmona	20368120-3	201773509-0	943218765	20

Estudiantes.

6.1. Ejemplos de consultas SQL.

```
SELECT * FROM Estudiantes;

Clemente Aguilar    20207131-k  201773580-3 988714567   20
Ignacio Aedo        19312897-4  201773531-4 912345678   20
Gonzalo Fernandez   19992851-1  201673557-4 987654321   21
Gabriel Carmona     20368120-3  201773509-0 943218765   20


SELECT Nota FROM Notas;

71
54
72
```

```
78
70
```

```
SELECT Nombre , Nota FROM Notas ORDER BY Nota DESC;

Gabriel Carmona     78
Gonzalo Fernandez   72
Clemente Aguilar    71
Gabriel Carmona     70
Ignacio Aedo        54
```

```
SELECT Telefono FROM Estudiantes WHERE Nombre='Clemente Aguilar';

988714567
```

```
SELECT Edad , Nota FROM Estudiantes INNER JOIN Notas
WHERE Estudiantes.Rol = Notas.Rol;

20 71
20 54
21 72
20 78
20 70
```

```
INSERT INTO Estudiantes (Nombre,Rut,Rol,Telefono,Edad)
VALUES ('Sebastian Godinez','19991933-8','201673501-1',988881234,21);

Se ha insertado 1 fila.
```

Nombre	Rut	Rol	Teléfono	Edad
Clemente Aguilar	20207131-k	201773580-3	988714567	20
Ignacio Aedo	19312897-4	201773531-4	912345678	20
Gonzalo Fernandez	19992851-1	201673557-4	987654321	21
Gabriel Carmona	20368120-3	201773509-0	943218765	20
Sebastian Godinez	19991933-8	201673501-1	988881234	21

Nota: La tabla expuesta es sólo para que se puedan apreciar los cambios realizados, el programa no se debe mostrar la tabla actualizada.

```
UPDATE Notas SET Nota=54 WHERE Rol='201673557-4';

Se ha actualizado 1 fila.
```

Nombre	Rol	Ramo	Sigla	Nota
Clemente Aguilar	201773580-3	Matematicas 1	Mat-021	71
Ignacio Aedo	201773531-4	Lenguajes de Programación	INF-253	54
Gonzalo Fernandez	201673557-4	Lenguajes de Programación	INF-253	54
Gabriel Carmona	201773509-0	Base de datos	INF-239	78
Gabriel Carmona	201773509-0	Sistemas Operativos	INF-246	70

```
INSERT INTO Estudiantes (Nombre) VALUES ('Fabio Pazos');
Se ha insertado 1 fila.
```

Nombre	Rut	Rol	Teléfono	Edad
Clemente Aguilar	20207131-k	201773580-3	988714567	20
Ignacio Aedo	19312897-4	201773531-4	912345678	20
Gonzalo Fernandez	19992851-1	201673557-4	987654321	21
Gabriel Carmona	20368120-3	201773509-0	943218765	20
Sebastian Godinez	19991933-8	201673501-1	988881234	21
Fabio Pazos				

```
INSERT INTO Estudiantes (Nombre, Rut) VALUES ('Fabio Pazos');
Error de Sintaxis !
SELECT  FROM Estudiantes
Error de Sintaxis !
SELECT Nombre FROM Estudiantes WHERE Rol='123456789-9';
La información solicitada no existe.
```

7. Consideraciones

- El nombre de las columnas se encuentra en la primera línea del archivo
- El programa debe ser capaz de mostrar la información obtenida a través de un SELECT utilizando la consola de comandos como medio.
- Para el caso del INSERT y el UPDATE, se debe modificar el archivo csv.
- No se pueden crear archivos auxiliares
- Las palabras claves como SELECT, UPDATE, WHERE , etc no pueden ser nombre de archivo ni de columna.
- En caso de que el nombre de una columna o archivo no exista, se debe señalar en un mensaje por consola su inexistencia.
- Se debe señalar con un mensaje por consola si el comando SQL ingresado es válido o no.
- Los archivo .csv serán las tablas y el nombres de estos sin la extensión es el nombre de la tabla.
- Se evaluará solo la sintaxis propuesta en este documento

8. Restricciones

Se permitirá el uso de librerías de manipulación de datos, sin embargo, no se permite el uso de funciones de lectura/escritura de archivos CSV, funciones de inserción, actualización de valores o uso de querys. Las funciones anteriores deben estar implementadas utilizando funciones nativas de Python y expresiones regulares.

En caso contrario, se evaluará el punto en cuestión como no implementado por el alumno.

9. Archivo a entregar

- **La entrega debe realizarse en un tar.gz y debe llevar el nombre: Tarea1LP_RolIntegrante-1_RolIntegrante-2.tar.gz**
- El tar.gz debe incluir dos archivos: el programa (el cuál debe llamarse sql.py) y un archivo README.txt.
- **El archivo README.txt debe contener nombre y rol de los integrantes del grupo e instrucciones detalladas para la correcta utilización de su programa.**

10. Sobre la Entrega

- Las funciones implementadas deben ser comentadas utilizando el siguiente formato:
  "
  Nombre de la función

  Entradas:
  (Tipo de dato) Descripción
  (Tipo de dato) Descripción
  ...

  Salida:
  (Tipo de dato) Output:

  Descripción de la función.
  "
- Se debe trabajar en grupos de dos personas.
- **El no cumplir con las reglas de entrega conlleva un descuento máximo de 30 puntos en su tarea.**
- La entrega será vía moodle y el plazo máximo de entrega es hasta el **Domingo 8 de Septiembre a las 23:55 hora moodle**.
- Por cada día de atraso se descontarán 20 puntos (10 puntos dentro de la primera hora).
- Las copias serán evaluadas con nota 0 y se informarán a las respectivas autoridades.
- No es obligatorio, pero se recomienda que utilicen GitHub para su trabajo grupal, es un repositorio en línea bastante útil y ampliamente utilizado. Es fácil aprender a utilizarlo sobre todo si se usa su aplicación para escritorio (La universidad tiene convenio con la página, pueden ir a buscar su mochila "gratis"
  `https://education.github.com/pack`).

11. Calificación

- Código (70 puntos)
  - Orden (5 puntos)
  - Expresiones Regulares (30 puntos)
  - Implementación de comandos SQL (35 puntos)
- Output (30 puntos)
  - Detección de errores (15 puntos)
  - Muestra y modifica los datos pedidos correctamente.(15 puntos)
- No usa expresiones regulares (-100 puntos)
- No respeta formato de entrega (-30 puntos MAX)