Entregable 3 - DSL

Una empresa de software quiere nacionalizar SQL con el objetivo de llegar al público desarrollador uruguayo sin requerir inglés como idioma, por lo que quiere diseñar y difundir el uso de USQL, conocido como *Uruguayan Structured Query Language*.

Para implementarlo, en una primera etapa se implementará por encima de SQL como un DSL en Python. Para la versión *beta*, se requerirá que las siguientes palabras clave sean transformadas a la sintaxis de USQL:

```
{
"SELECT": "TRAEME",
"*": "TODO",
"FROM": "DE LA TABLA",
"WHERE": "DONDE",
"GROUP BY": "AGRUPANDO POR",
"JOIN": "MEZCLANDO",
"ON": "EN",
"DISTINCT": "LOS DISTINTOS",
"COUNT": "CONTANDO",
"INSERT INTO": "METE EN",
"VALUES": "LOS VALORES",
"UPDATE": "ACTUALIZA",
"SET": "SETEA",
"DELETE FROM":
"BORRA DE LA",
"ORDER BY": "ORDENA POR",
"LIMIT": "COMO MUCHO",
"HAVING": "WHERE DEL GROUP BY",
"EXISTS": "EXISTE",
"IN": "EN ESTO:",
"BETWEEN": "ENTRE",
"LIKE": "PARECIDO A",
"IS NULL": "ES NULO",
"ALTER TABLE": "CAMBIA LA TABLA",
"ADD COLUMN": "AGREGA LA COLUMNA",
"DROP COLUMN": "ELIMINA LA COLUMNA",
"CREATE TABLE": "CREA LA TABLA",
"DROP TABLE": "TIRA LA TABLA",
"DEFAULT": "POR DEFECTO",
"UNIQUE": "UNICO",
"PRIMARY KEY": "CLAVE PRIMA",
"FOREIGN KEY": "CLAVE REFERENTE",
"NOT NULL": "NO NULO",
"CAST": "TRANSFORMA A",
}
```

Deben implementarse:

1) Un DSL utilizando la biblioteca PLY de Python que permita traducir consultas de SQL a USQL y viceversa. Se debe verificar mediante tests unitarios que la función sea capaz de manejar estas consultas correctamente, no solo traduciendo la sintaxis, sino verificando que al traducir a SQL, la consulta sea válida.
2) Un Fluent API en Python que permita realizar consultas, o tomar consultas parciales ya existentes y agregar filtros encadenándolos tantas veces como se desee.

El lenguaje creado debe soportar **como mínimo** las siguientes consultas:

```
TRAEME TODO DE LA TABLA usuarios DONDE edad > 18;

TRAEME LOS DISTINTOS nombre DE LA TABLA clientes DONDE ciudad =
'Madrid';

METE EN usuarios (nombre, edad) LOS VALORES ('Juan', 25);

ACTUALIZA empleados SETEA salario = 3000 DONDE puesto = 'ingeniero';

TRAEME TODO DE LA TABLA pedidos MEZCLANDO clientes EN pedidos.cliente_id
= clientes.id DONDE clientes.ciudad = 'Barcelona';

TRAEME CONTANDO(TODO) DE LA TABLA ventas AGRUPANDO POR producto WHERE
DEL GROUP BY COUNT(*) > 5;

BORRA DE LA tabla clientes DONDE edad ENTRE 18 Y 25;

CAMBIA LA TABLA empleados AGREGA LA COLUMNA direccion VARCHAR(255) NO
NULO;

CAMBIA LA TABLA empleados ELIMINA LA COLUMNA direccion;
```

Consideraciones:

- Puede asumir que si hay algún token o palabra clave que no se menciona en la lista *beta*, mantendrá su valor original en la sintaxis SQL, o puede tomar un valor que considere adecuado. Cree una base de datos local para realizar pruebas iniciales.
- Se deberá realizar tests unitarios para todo el código, teniendo una cobertura de al menos 85%. Investigue sobre el paquete *pycobertura* para ello.
- La entrega deberá contener el código fuente y un muy breve informe de la cobertura lograda.
- Deberá implementarse el manejo de excepciones para consultas no válidas.

La entrega se realizará vía *moodle*, siendo la fecha límite de entrega el domingo 27 de octubre a las 23:59.