Proyecto de diseño de compiladores

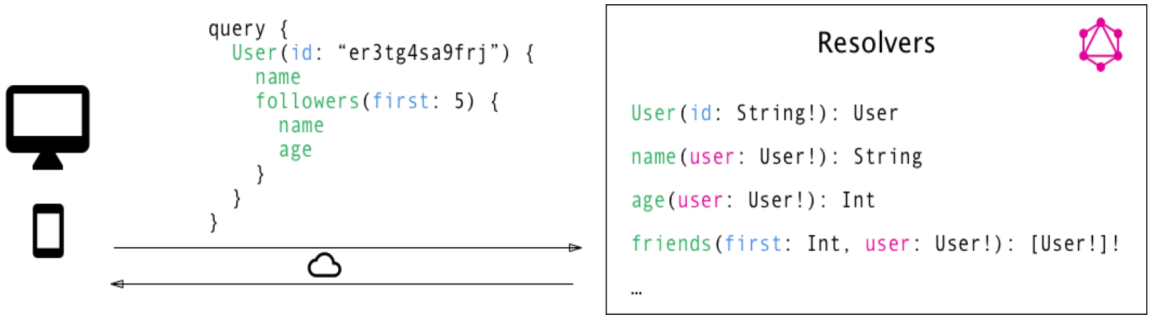
Su proyecto consistirá en la creación de un DSL para interpretar el lenguaje GraphQL y se ejecute en SQLServer, es decir, que las consultas se hagan al estilo GraphQL pero considerando las relaciones en SQLServer. Para ello deberán realizar un servidor GraphQL (caso 1 de su arquitectura ver video de arquitectura en https://www.howtographql.com/basics/3-big-picture/) pero que se limite únicamente a la parte correspondiente a las consultas (nada de mutaciones y subscripciones).

Lo siguiente es tomado del curso de GraphQL de EDX (ver URL abajo):

In the first and third use case where the server is connected to a database, data fetching from the database can happen in two ways:

1. Resolver functions
2. Compiler approach.

In the resolver functions approach, the incoming GraphQL query fields are mapped to functions which fetch the necessary data. This approach can lead to multiple hits to the database for nested fields.



For example, in the above screenshot, resolvers are written for the user's name and their followers, and each resolver will hit the database individually.

In the compiler approach, the GraphQL query is parsed into a representation of GraphQL AST. The GraphQL AST can now be converted into a query language that the database understands (SQL, NoSQL etc).

```
GraphQL Parser -> GraphQL AST -> SQL AST -> SQL
```

This approach ensures that there is only a single hit to the database, however nested the GraphQL query may be. Open source GraphQL Engines like Hasura and PostGraphile implement this approach where the GraphQL query is compiled into a single SQL query

El proyecto se basa en las siguientes fases:

Fase 1:

1. Realizar una búsqueda y resumen de lo que es GraphQL

Proyecto de diseño de compiladores

2. Implementar su parte léxica en ANTLR4

   En realidad, hay alguien que ya hizo algo de GraphQL y ANTLR, deberán verificarlo y explicarlo a través de un documento y deberán cambiar las variables léxicas a todas mayúsculas como se estila en ANTLR.

Fase 2:

Para la segunda fase, realizarán en ANTLR la parte sintáctica

En realidad, hay alguien que ya hizo algo de GraphQL y ANTLR, por lo que deberán explicar la parte sintáctica y faltaría las otras partes (semántica, traducción y ejecución sobre SQLServer).

Fase 3:

Parte semántica, traducción y ejecución sobre SQLServer

Realización del artículo el cuál deberá contener las siguientes secciones y presentación final:

1) Autores y filiaciones
2) Resumen
3) Introducción
4) Estado del arte.
5) Desarrollo.
6) Resultados obtenidos
7) Conclusiones y trabajo futuro
8) Bibliografía. Es muy importante citar a todo lo que le da soporte a su discurso. No pongan una lista de referencias de las cuales no se encuentren citadas en su discurso

Algunas URLS de ayuda:

https://graphql.org/learn/
https://www.edx.org/course/exploring-graphql-a-query-language-for-apis

Se cuenta con alguien que ya hizo parte de la chamba y habrá que verfificarlo y explicarlo:

https://github.com/antlr/grammars-v4/tree/master/graphql
https://github.com/arakelian/graphql-parser
https://github.com/charithe/antlr4-graphql/blob/master/GraphQL.g4

Tecnológico de Monterrey

Proyecto de diseño de compiladores

Y posiblemente otras

Sobre la utilización de GraphQL en bases de datos SQL:

https://productionreadygraphql.com/blog/2020-05-21-graphql-to-sql

Ahí mencionan la de Hasura (de acuerdo al curso de EDX de GraphQL), Hasura is an open source engine that connects to your databases & microservices and auto-generates a production-ready GraphQL backend. Hasura gives you realtime GraphQL APIs that are high-performance, scalable, extensible & secure (with authorization baked in) sobre Postgres y SQL Server (bueno por realizares)