Carlos Ariel Rubio Espinosa

Creación de Schema:

<country>: string @index(term) .

<dgraph.drop.op>: string .

<dgraph.graphql.p_query>: string @index(sha256) .

<dgraph.graphql.schema>: string .

<dgraph.graphql.xid>: string @index(exact) @upsert .

<dgraph.namespace.id>: int @index(int) @upsert .

<dgraph.namespace.name>: string @index(exact) @upsert .

<duration>: int .

<genre>: string @index(hash) .

<has_song>: uid @reverse .

<label>: string @index(hash) .

<name>: string @index(exact, fulltext) .

<part_of>: uid @reverse .

<release_year>: int @index(int) .

<sung_by>: uid @reverse .

<title>: string .

type <dgraph.graphql> {

dgraph.graphql.schema

dgraph.graphql.xid

}

type <dgraph.graphql.persisted_query> {

dgraph.graphql.p_query

}

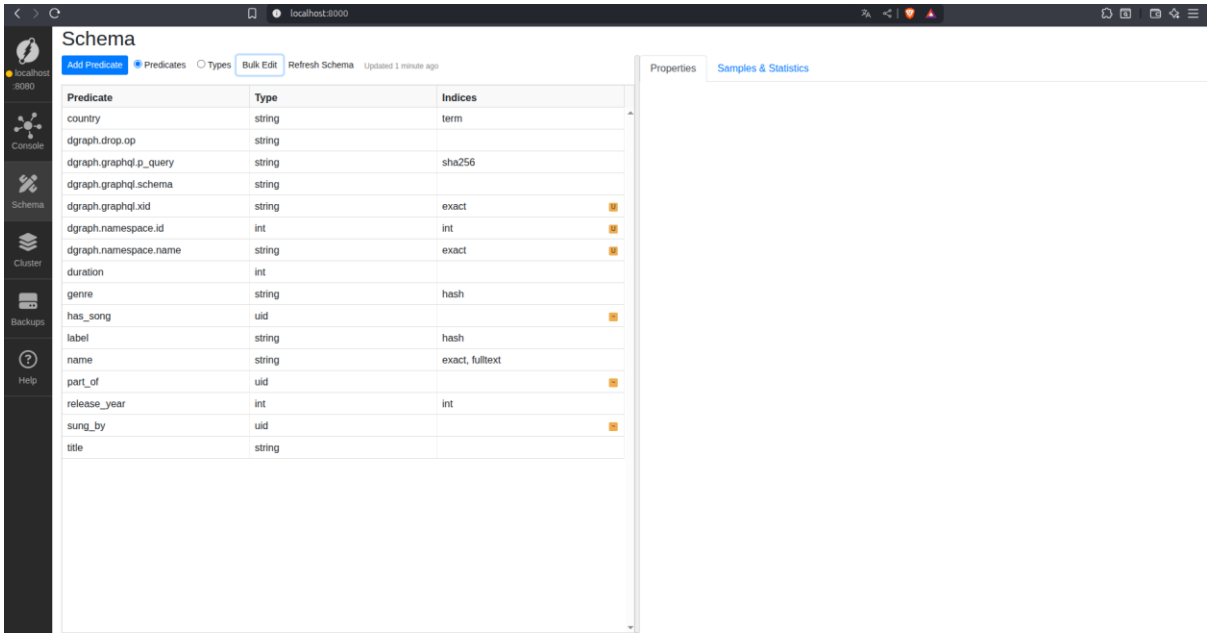
type <dgraph.namespace> {

dgraph.namespace.name

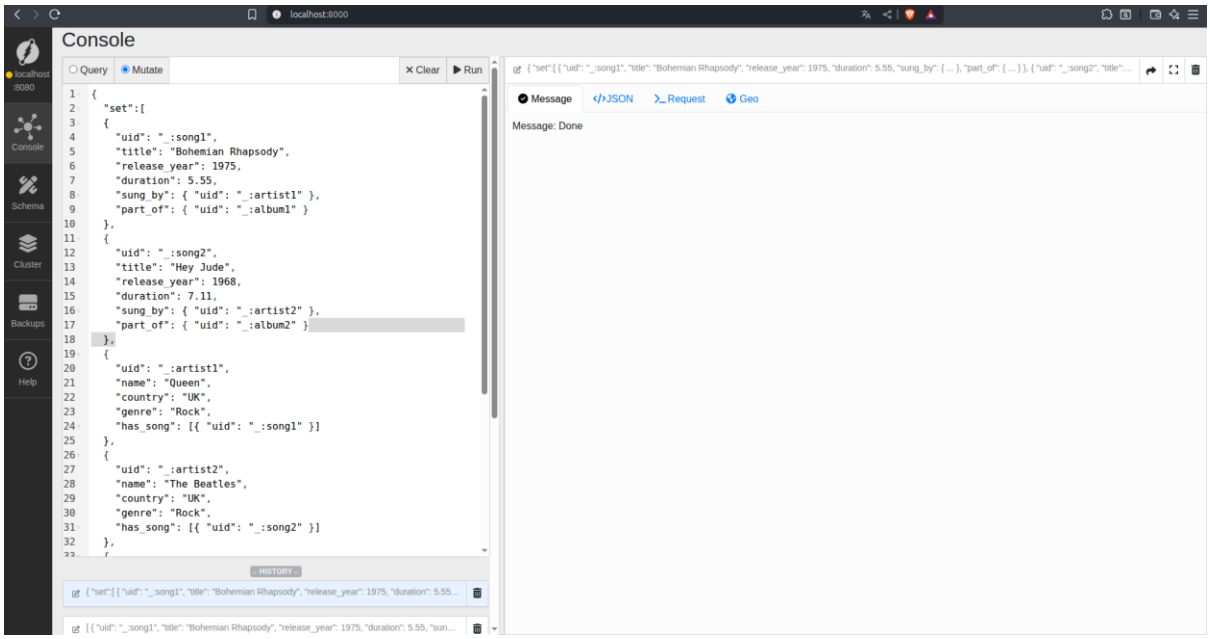
dgraph.namespace.id

}

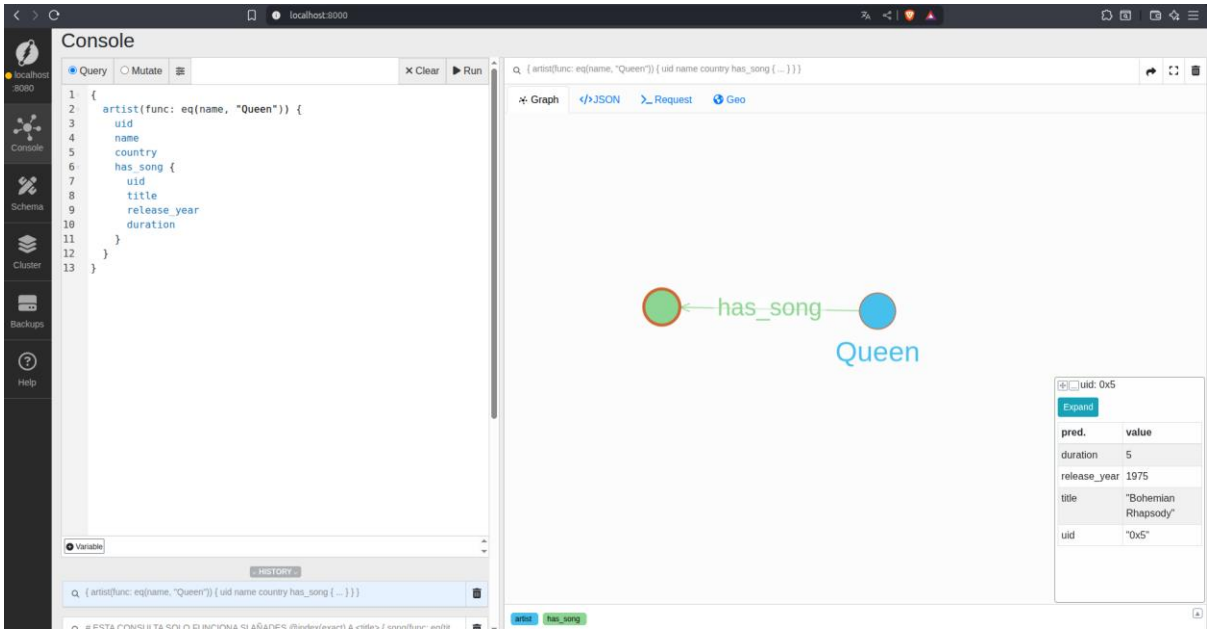
Prueba del Schema hecho:



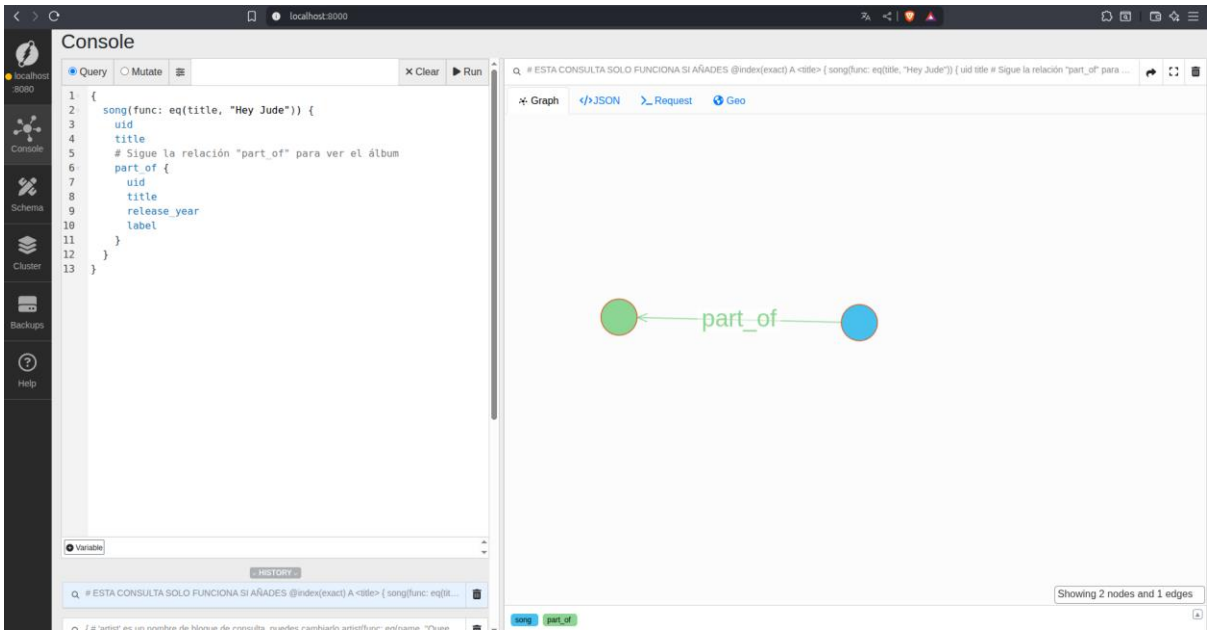
Inserción de datos:



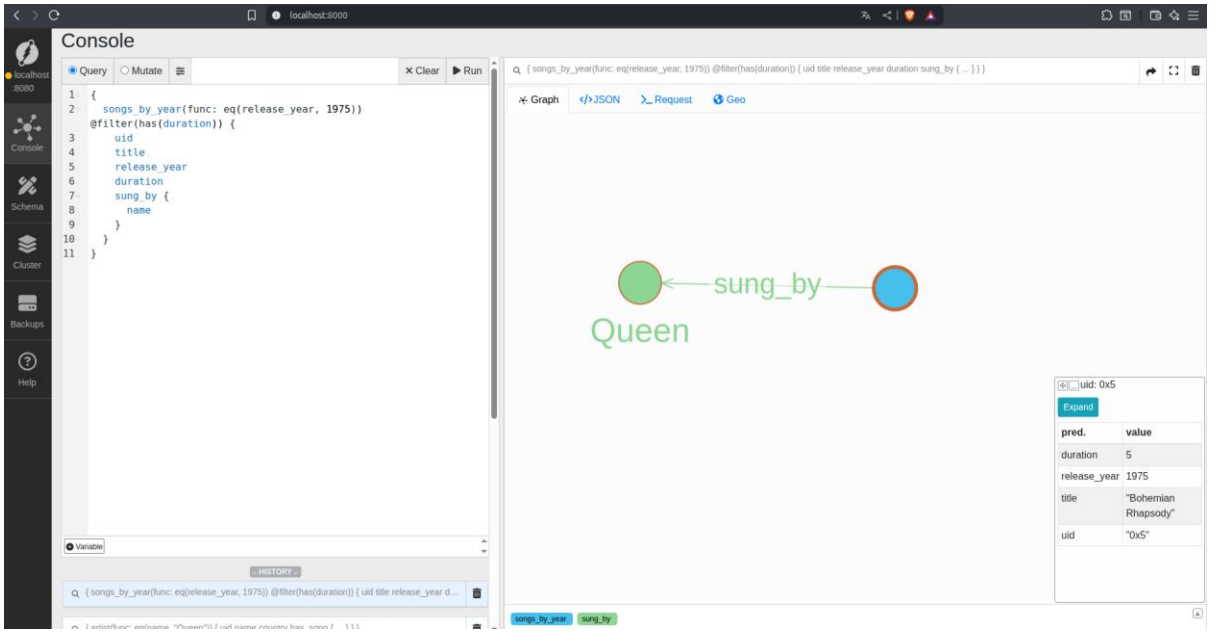
Primera Consulta:



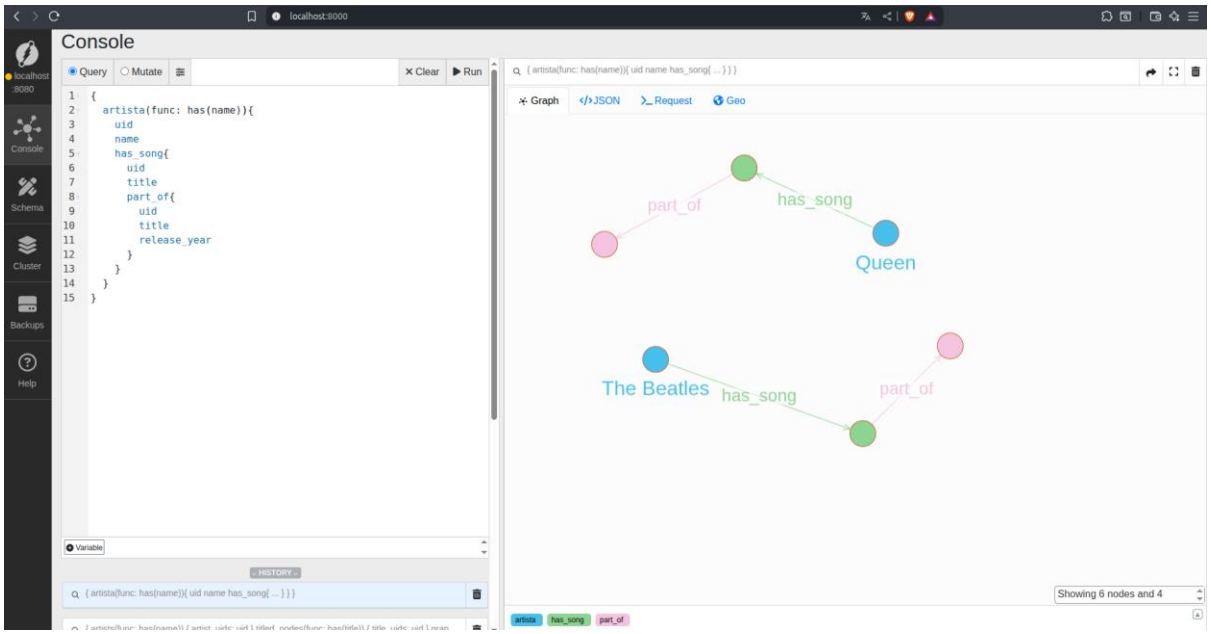
Segunda Consulta:



Tercera Consulta:



Mostrar todos los datos:



Conclusión:

Al ser esto una creación y consulta más guiada, siento que es más fácil de entender cómo funciona Dgraph y más rápida de hacer esto que el primer laboratorio.