THE RESOLVE FUNCTION

In this GraphQL tutorial I'll show you how we can use the resolve function to go out and actually grab our data for the GraphQL server to return to the front-end.

schema.js

Insert code

```
const graphql = require('graphql');

const { GraphQLObjectType, GraphQLString, GraphQLSchema } = graphql;

// dummy data
var books = [
    { name: 'Name of the Wind', genre: 'Fantasy', id: '1' },
    { name: 'The Final Empire', genre: 'Fantasy', id: '2' },
    { name: 'The Long Earth', genre: 'Sci-Fi', id: '3' },
];

const BookType = new GraphQLObjectType({
...
});

const RootQuery = new GraphQLObjectType({
...
});

module.exports = new GraphQLSchema({
    query: RootQuery
});
```

Install Lodash

In the terminal write next command

- npm i lodash

schema.js

```
const graphql = require('graphql');
const _ = require('lodash');

const { GraphQLObjectType, GraphQLString, GraphQLSchema } = graphql;

// dummy data
var books = [
...
];
const BookType = new GraphQLObjectType({
...
});
const RootQuery = new GraphQLObjectType({
    name: 'RootQueryType',
    fields: {
        book: {
            type: BookType,
            args: { id: { type: GraphQLString } },
            resolve(parent, args){
                // code to get data from db / other source
                return _.find(books, { id: args.id });
            }
        }
    }
});

module.exports = new GraphQLSchema({
    query: RootQuery
});
```

Insert code