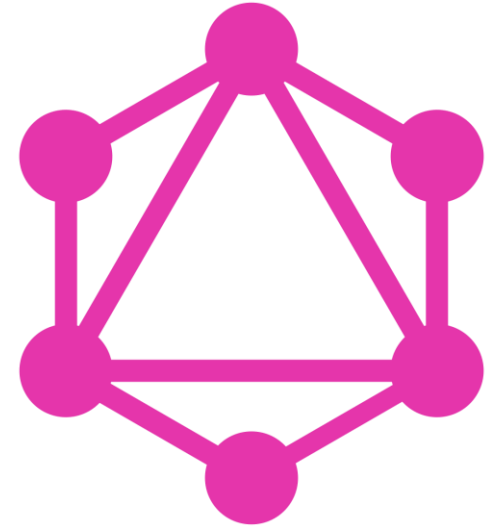


GraphQL in .NET Core



- *Poornima Nayar*
- *Freelance .NET Developer*
- *Berkshire, UK*
- *Microsoft MVP, Umbraco MVP*
- *Mummy, Reading, Carnatic Music*
- *@poornimanayar*



<https://poornimanayar.co.uk>

GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data.

*Specify **types** and ask for specific **fields** on those
types*

*GraphQL is a **specification***

*GraphQL lets you traverse the data graph
and extract parts of it*



/user/id

/user/id/events

/user/id/friends-suggestions

/user/id/friends-birthdays

Over fetching

Under fetching

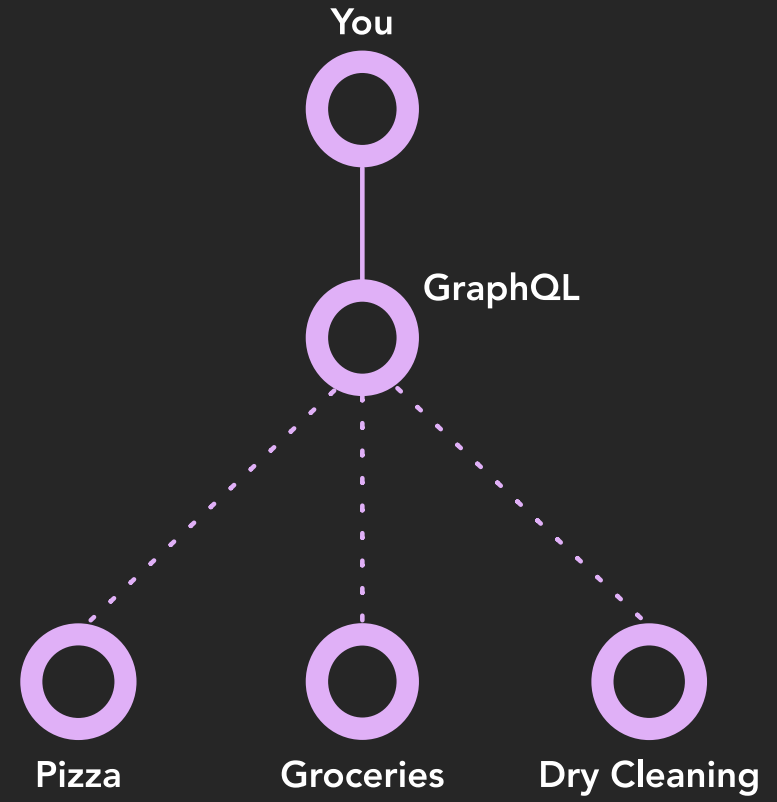
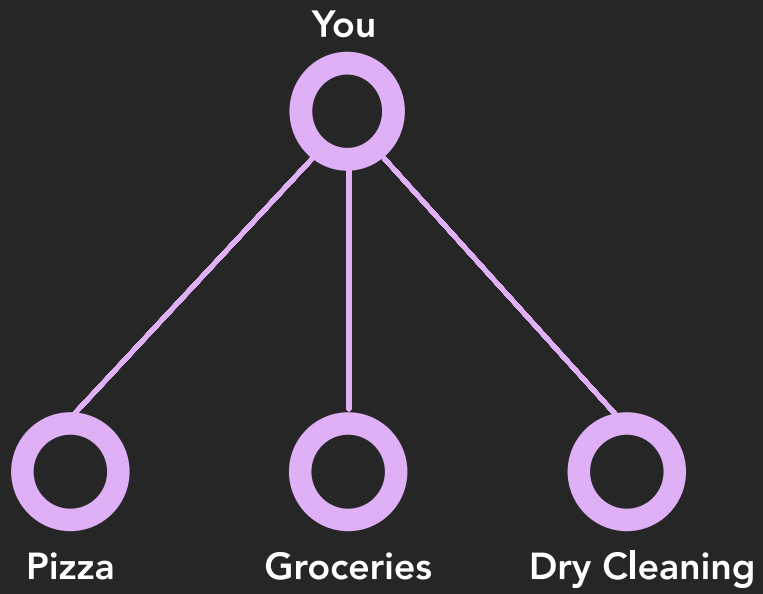
N+1

*Single, smart endpoint that takes in complex queries and
builds the data output*

The clients specify the shape of the data that they need, and the server responds back with the exact same data graph as the query.

```
query{
  course(id:1045){
    id,
    credits,
    title,
    enrollments{
      student{
        firstName,
        lastName,
        emailAddress
      }
    }
  }
}
```

```
{
  "data": {
    "course": {
      "id": 1045,
      "credits": 4,
      "title": "Calculus",
      "enrollments": [
        {
          "student": {
            "firstName": "Meredith",
            "lastName": "Alonso",
            "emailAddress": "Meredith@graphqlschool.com"
          }
        },
        {
          "student": {
            "firstName": "Peggy",
            "lastName": "Justice",
            "emailAddress": "Peggy@graphqlschool.com"
          }
        }
      ]
    }
  },
  "extensions": {↔}
}
```



- Strongly Typed
- Specification
- Introspective
- Hierarchical
- JSON response that mirrors the query
- Application Layer
- Version Free
- Transport-layer agnostic



THE GRAPHQL TYPE SYSTEM

*GraphQL schema is specified using **GraphQL SDL (Schema Definition Language)**, also sometimes referred to as just **GraphQL Schema Language**.*

Schema

Object Types

*Representation of
object*

Queries

Get data

Mutations

*Insert/Update/
Delete data*

Subscriptions

Pushed Updates

TYPES

Field

Name

Name of the field

Scalar/Complex
Types

Type of the field

Resolvers

*How to resolve the
field to concrete
data*

Anatomy of a GraphQL Type

```
type StudentType {  
  id: Int!  
  firstName: String!  
  lastName: String!  
  emailAddress: String!  
  enrollmentDate: DateTime!  
  enrollments: [EnrollmentType]  
}
```

Fields

Type

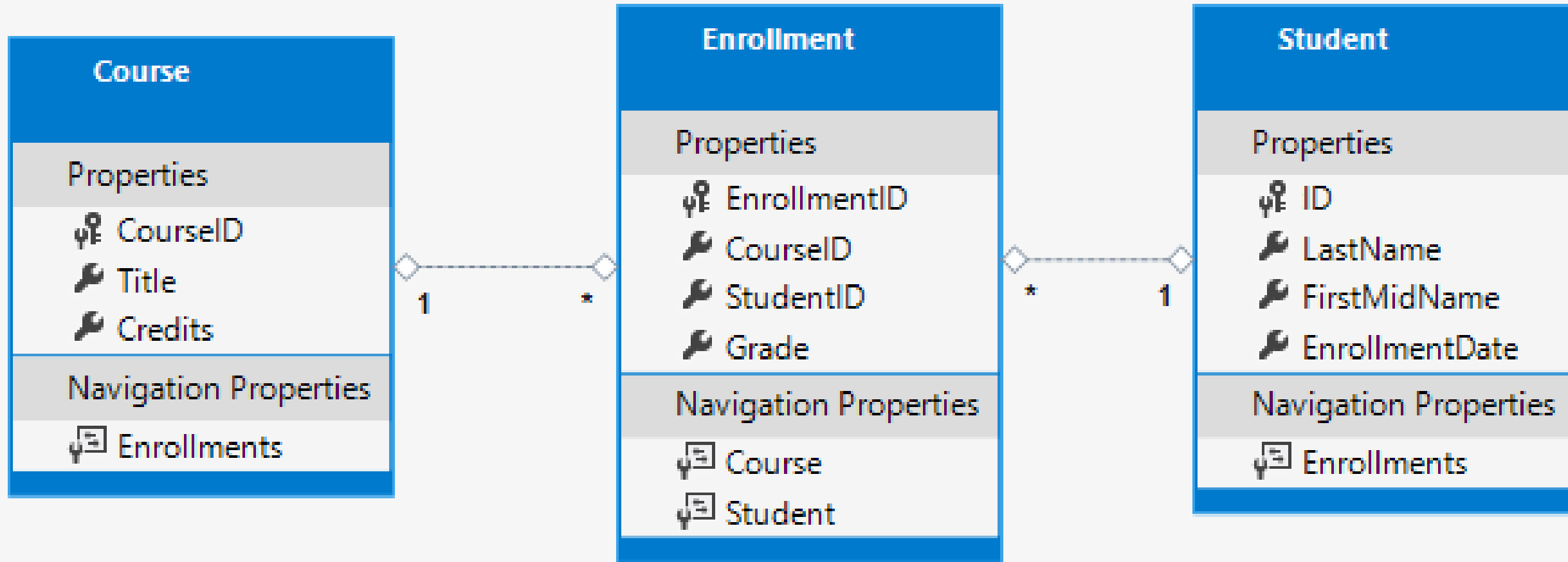
Scalars / Complex type

- Int
- Float
- String
- Boolean
- ID

Enumeration Types

Non-nullable fields `id: Int!`

Lists `enrollments: [EnrollmentType]`



GraphQL in .NET Core

- GraphQL.NET
- HotChocolate

- GraphQL.Net Client
- StrawberryShake

IDE

- GraphiQL
- GraphQL Playground
- Insomnia
- Altair
- Firecamp
- Voyager
- BananaCakePop

- Syntax Highlighting
- Intellisense
- Real-time error highlighting and reporting
- Support for variables and auth tokens
- Run and inspect query results
- Schema introspection, up-to-date documentation and help!

GET STARTED

GraphQL Endpoint

Install-Package GraphQL

Install-Package GraphQL.Server.Transports.AspNetCore

Install-Package GraphQL.Server.Transports.AspNetCore.SystemTextJson

Install-Package GraphQL.Server.Ui.Playground

Install-Package GraphQL.Server.Ui.GraphiQL

Client App

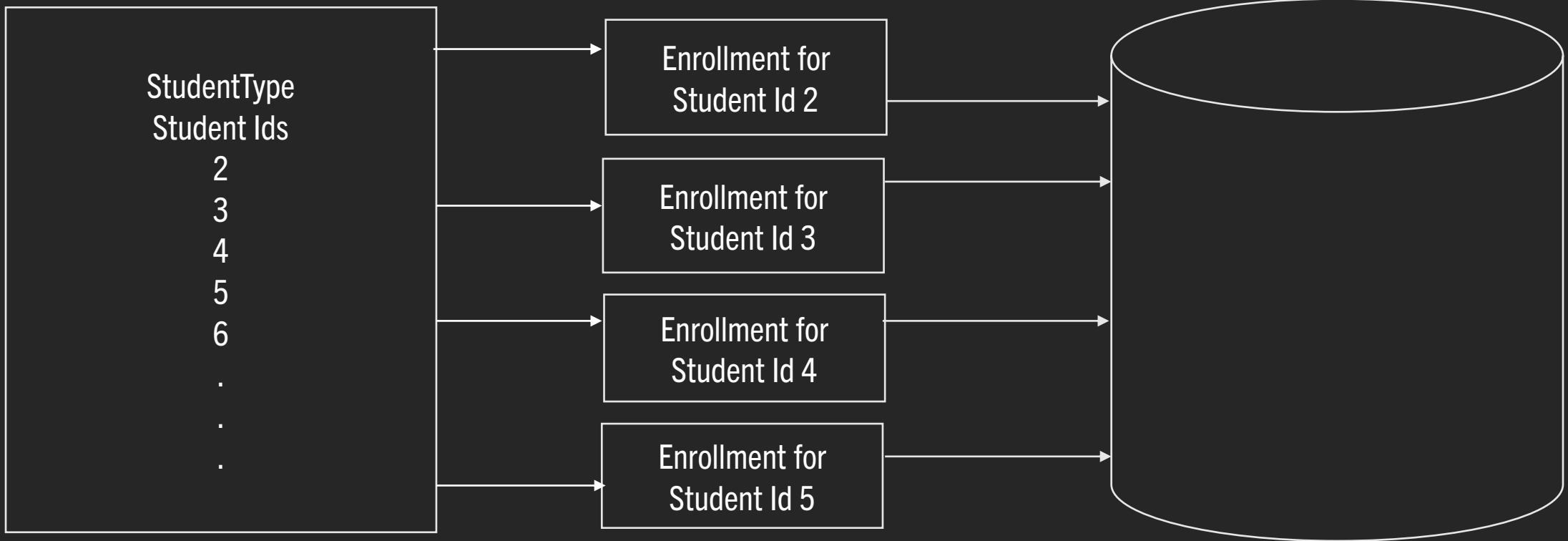
Install-Package GraphQL.Client

Install-Package GraphQL.Client.Serializer.SystemTextJson

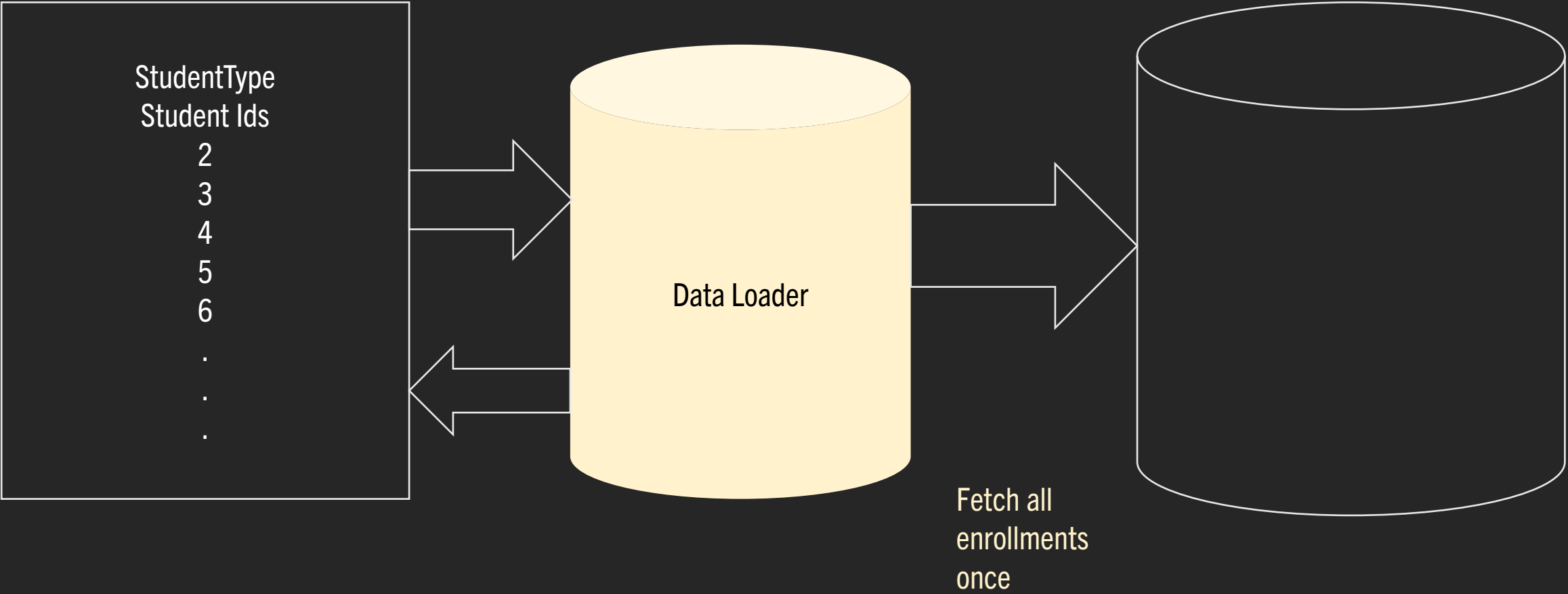


QUERY

- One of the operation types in GraphQL
- Top-level entry point for read operations
- Get/fetch the data, ask for specific fields on objects using a POST operation
- `query` keyword, although not needed, best practice
- Hierarchical JSON response that mirrors the query
- Queries can be executed in parallel



N+1 issue



MUTATIONS



- Top-level entry point for write operations
- HTTP POST
- `mutation` keyword
- Accepts a `InputType` as an argument
- Returns a `GraphType` as the result which can be queried
- Mutations are executed in series, one-by-one

SUBSCRIPTIONS



- Updates are pushed from the server, not polled by client
- Clients can choose to listen
- Works over WebSockets
- HTTP POST
- subscription keyword
- Subscriptions are stateful

Install-Package GraphQL.Server.Transports.Subscriptions.WebSockets

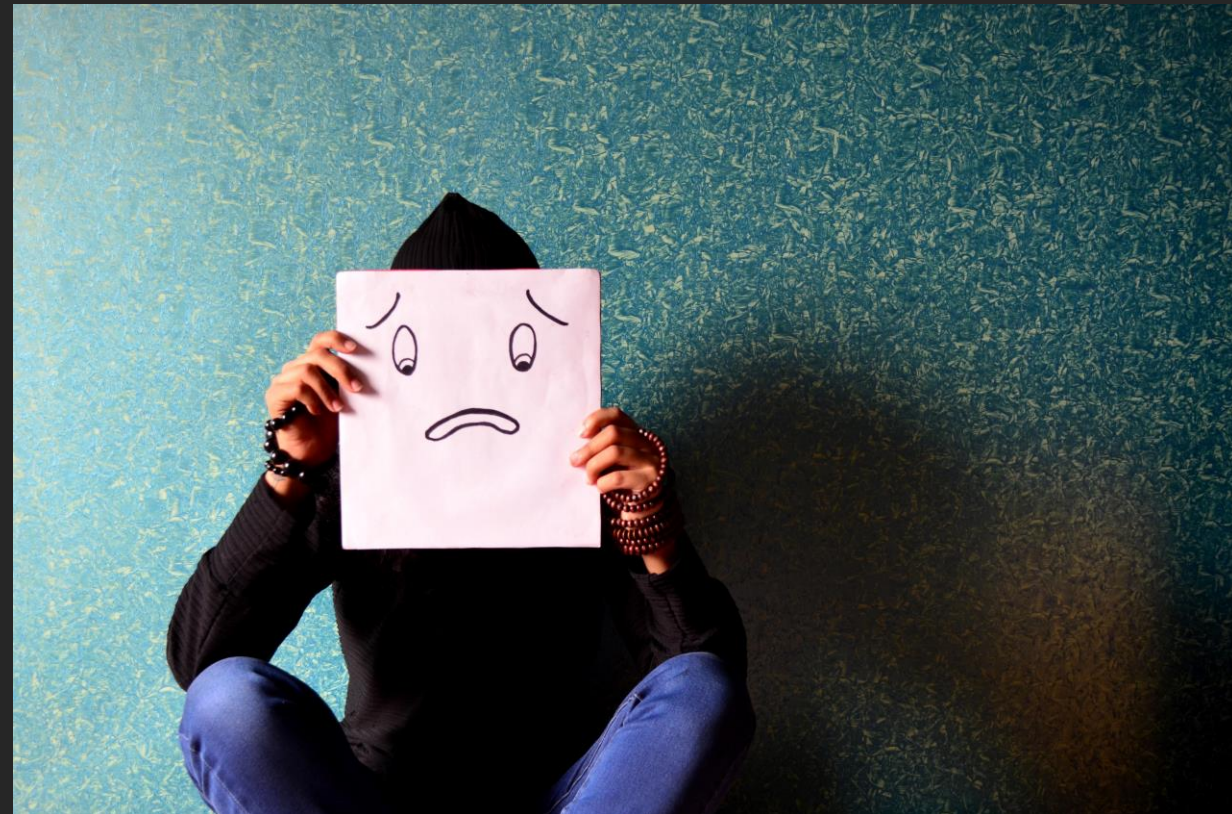
Install-Package GraphQL.Client.Abstractions.Websocket

- Pagination
- Filtering & Sorting
- Authorization
- Caching
- Automated Persisted Queries
- Microservice-style architecture

- Exact Fetching
- Loosely coupled client and server
- Autogenerating API documentation
- Version Free
- Ability to lock down fields by permission



- Caching complexity
- File uploading
- Nested queries could result in bringing out the entire database
- Learning curve



<http://bit.ly/graphql-in-netcore>

