THAM KHẢO NHANH GRAPHQL

Schemas, queries, mutations, types, fragments

Định Nghĩa Schema

Schema Root

```
schema {
  query: Query
  mutation: Mutation
}
```

Object Type

```
type User {
  id: ID!
  name: String!
  email: String
}
```

Queries

Query Cơ Bản

```
query {
  user(id: "1") {
    name
    email
  }
}
```

Query Có Tên với Alias

```
query GetUsers {
  admin: user(role: ADMIN) { name }
  guest: user(role: GUEST) { name }
}
```

Mutations

Mutation Cơ Bản

```
mutation {
  createUser(input: { name: "Alice" }) {
    id
    name
  }
}
```

Input Types

```
input CreateUserInput {
  name: String!
  email: String
}
```

Subscriptions

Subscription Cơ Bản

```
subscription {
  messageAdded(channel: "general") {
    text
    sender { name }
  }
}
```

Tổng Quan

`subscription`	Dữ liệu thời gian thực qua WebSocket
`Server push`	Server gửi cập nhật đến client
`Single field`	Chỉ một trường root mỗi subscription

Types

Scalar Types

`Int`	Số nguyên có dấu 32-bit
`Float`	Số thực dấu phẩy động độ chính xác kép
`String`	Chuỗi ký tự UTF-8
`Boolean`	true hoặc false
`ID`	Định danh duy nhất (serialize như String)

Bộ Chỉnh Kiểu

`String`	Chuỗi có thể null
`String!`	Chuỗi không được null
`[String]`	Danh sách nullable gồm các chuỗi nullable
`[String!]!`	Danh sách không-null gồm các chuỗi không-null

Enum & Union

```
enum Role { ADMIN USER GUEST }
union SearchResult = User | Post
interface Node { id: ID! }
```

Arguments & Variables

Variables

```
query GetUser($id: ID!) {
  user(id: $id) {
    name
  }
}
# Variables: { "id": "123" }
```

Giá Trị Mặc Định

```
query GetUsers($limit: Int = 10) {
  users(limit: $limit) { name }
}
```

Fragments

Fragment Có Tên

```
fragment UserFields on User {
  id
  name
  email
}
```

Sử Dụng Fragments

```
query {
  user(id: "1") { ...UserFields }
  admin: user(id: "2") { ...UserFields }
}
```

Inline Fragment

```
query {
  search(text: "a") {
    ... on User { name }
    ... on Post { title }
  }
}
```

Directives

Directives Tích Hợp Sẵn

`@include(if: Boolean!)`	Chỉ bao gồm trường khi điều kiện đúng
`@skip(if: Boolean!)`	Bỏ qua trường khi điều kiện đúng
`@deprecated(reason: String)`	Đánh dấu trường là deprecated

Ví Dụ Sử Dụng

```
query GetUser($withEmail: Boolean!) {
  user(id: "1") {
    name
    email @include(if: $withEmail)
  }
}
```

Introspection

Introspection Kiểu

```
query {
  __type(name: "User") {
    name
    fields { name type { name } }
  }
}
```

Introspection Schema

```
query {
  __schema {
    types { name kind }
    queryType { name }
  }
}
```

Trường Introspection

`__schema`	Truy vấn các kiểu và directive của schema
`__type(name:)`	Truy vấn kiểu cụ thể theo tên
`__typename`	Trả về tên kiểu của bất kỳ đối tượng nào

Mẫu Phổ Biến

Phân Trang (Relay-style)

```
query {
  users(first: 10, after: "cursor") {
    edges { node { name } cursor }
    pageInfo { hasNextPage }
  }
}
```

Xử Lý Lỗi

```
{
  "data": { "user": null },
  "errors": [{ "message": "Not found",
    "path": ["user"] }]
}
```

Thực Hành Tốt

`Name operations`	Luôn đặt tên cho query và mutation
`Use variables`	Không nội suy giá trị vào chuỗi query
`Request only needed fields`	Tránh over-fetching bằng selection chính xác
`Use fragments`	Chia sẻ tập trường giữa các query