

Get All Products with associated category (one to one relation)

The screenshot displays the GraphQL Playground interface. On the left, the 'Documentation' sidebar shows the 'Query' section with a list of fields: 'categories: [Category]', 'category(...): Category', 'product(...): Product', and 'products: [Product]'. The 'products' field is selected. The main editor shows a query named 'AllProducts' with the following structure:

```
query AllProducts {
  products {
    code
    title
    active
    price
    stock
    category {
      code
      name
      categoryUrl
      active
    }
  }
}
```

Below the query editor, the 'Variables' tab is active, showing an empty JSON object: {}.

On the right, the 'Response' tab shows the JSON output of the query:

```
{
  "data": {
    "products": [
      {
        "code": "1000",
        "title": "Samsung Mobile M56",
        "active": true,
        "price": 15999.99,
        "stock": 100,
        "category": {
          "code": "5000",
          "name": "Smartphone",
          "categoryUrl": "smartphone",
          "active": true
        }
      },
      {
        "code": "1001",
        "title": "Samsung Galaxy S22",
        "active": true,
        "price": 65999.99,
        "stock": 10,
        "category": {
          "code": "5000",
          "name": "Smartphone",
          "categoryUrl": "smartphone",
          "active": true
        }
      },
      {
        "code": "1002",
        "title": "ThinkBook 16 40.64cms",
        "active": true,
        "price": 65999.99,
        "stock": 25,
        "category": {
          "code": "5001",
          "name": "Laptop",
          "categoryUrl": "laptop",
          "active": true
        }
      }
    ]
  }
}
```

Get All Categories:

The screenshot displays the GraphQL Playground interface. At the top, the URL bar shows 'http://localhost:19001/graphql'. The left sidebar contains 'Documentation' and 'Fields' sections. The 'Fields' section lists the available types: 'categories: [Category]', 'category(...): Category', 'product(...): Product', and 'products: [Product]'. The main area is divided into three tabs: 'Operation', 'Variables', and 'Post-Operation Script'. The 'Operation' tab is active, showing a query:

```
query AllCategories{
  categories {
    code
    name
    categoryUrl
    active
  }
}
```

. The 'Variables' tab is also visible, showing an empty JSON object:

```
{}
```

. The 'Post-Operation Script' tab is empty. On the right, the 'Response' tab shows the JSON output:

```
{
  "data": {
    "categories": [
      {
        "code": "5000",
        "name": "Smartphone",
        "categoryUrl": "smartphone",
        "active": true
      },
      {
        "code": "5001",
        "name": "Laptop",
        "categoryUrl": "laptop",
        "active": true
      },
      {
        "code": "5002",
        "name": "TV",
        "categoryUrl": "tv",
        "active": true
      },
      {
        "code": "5003",
        "name": "LED TV",
        "categoryUrl": "ledtv",
        "active": true
      },
      {
        "code": "5004",
        "name": "Camera",
        "categoryUrl": "/camera",
        "active": true
      },
      {
        "code": "5005",
        "name": "Smartwatch",
        "categoryUrl": "smartwatch",
        "active": true
      }
    ]
  }
}
```

. The status bar at the bottom right indicates 'STATUS 200 | 30.0ms | 879B'.

Query – find Product By Code

The screenshot displays the GraphQL Playground interface. On the left, the 'Documentation' panel shows the schema for the 'product' type, including fields like 'code', 'active', 'category', 'description', 'image', 'price', 'stock', and 'title'. The main editor shows a query:

```
query Product {
  product(code: "1000") {
    code
    title
    price
    stock
    active
    category {
      code
      name
      categoryUrl
      active
    }
  }
}
```

. The 'Response' panel on the right shows the JSON output:

```
{
  "data": {
    "product": {
      "code": "1000",
      "title": "Samsung Mobile M56",
      "price": 15999.99,
      "stock": 100,
      "active": true,
      "category": {
        "code": "5000",
        "name": "Smartphone",
        "categoryUrl": "smartphone",
        "active": true
      }
    }
  }
}
```

. The bottom panel shows the 'Variables' tab with a single variable 'code' set to '1000'.

Query – find Category By Code

The screenshot displays the GraphQL Playground interface. The top bar shows the URL `http://localhost:19001/graphql` and a `Publish` button. The left sidebar contains the `Documentation` panel, which shows the `category` type with fields `code`, `name`, `categoryUrl`, and `active`. The main area is divided into three tabs: `Operation`, `Response`, and `Variables`. The `Operation` tab is active, showing a query that fetches the `category` for a given `code`. The `Response` tab shows the JSON response for the query. The `Variables` tab is also visible, showing the input variables for the query.

```
query Product {
  category(code: "5003") {
    code
    name
    categoryUrl
    active
  }
}
```

```
{
  "data": {
    "category": {
      "code": "5003",
      "name": "LED TV",
      "categoryUrl": "ledtv",
      "active": true
    }
  }
}
```

Mutation - createCategory() – Code 5013

SANDBOXhttp://localhost:19001/grajPublish

<<createCategory ×+

Documentation

Root › Mutation › createCategory

← createCategory: Category ✓

Arguments

- ✓ name: String!
- ✓ categoryUrl: String!
- ✓ active: Boolean!

Fields ↓ − ∨

- ✓ active: Boolean
- ✓ categoryUrl: String
- ⊕ code: ID
- ✓ name: String
- ⊕ parentId: String

Operation

↑📄∨▶ createCategory

```
1 mutation createCategory {
2   | createCategory(name: "Air Conditioner", categoryUrl: "/airconditioner", active: true) { ...
3     | name
4     | active
5     | categoryUrl
6   }
7 }
```

VariablesHeadersPre-Operation ScriptPost-Operation Script

1JSON

Respo... ▾☰🔍STATUS 200 | 29.0ms | 101B

```
{
  "data": {
    "createCategory": {
      "name": "Air Conditioner",
      "active": true,
      "categoryUrl": "/airconditioner"
    }
  }
}
```

Find Category (code : 5013)

The screenshot displays the GraphQL Playground interface. At the top, the URL bar shows 'http://localhost:19001/graphql' and a 'Publish' button. The left sidebar contains 'Documentation' and 'Root > Query > category' with a 'category: Category' argument. The main area shows a query:

```
query findCategory {
  category(code: "5013") {
    code
    name
    categoryUrl
    active
    __typename
  }
}
```

 Below the query is an 'Operation' tab with the same query. The right sidebar shows the 'Response' in JSON format:

```
{
  "data": {
    "category": {
      "code": "5013",
      "name": "Air Conditioner",
      "categoryUrl": "/airconditioner",
      "active": true,
      "__typename": "Category"
    }
  }
}
```

 At the bottom, there are tabs for 'Variables', 'Headers', 'Pre-Operation Script', and 'Post-Operation Script', with 'Variables' currently selected and showing an empty field.