1.

datamodel.prisma tables

scalar DateTime

type Category {

id: ID! @unique

categoryname: String

}

type Product {

id: ID! @unique

category: Category!

title: String!

actor: String!

price: Float!

special: Int!

common\_prod\_id: Int!

}

type Inventory {

id: ID! @unique

product: Product!

quan\_in\_stock: Int!

sales: Int!

}

type Reorder {

id: ID! @unique

product: Product!

date\_low: DateTime!

quan\_low: Int!

date\_reordered: DateTime!

quan\_reordered: Int!

date\_expected: DateTime!

}

2.

Query Resolvers

const resolvers = {

Query: {

Querycategories(root, args, context){

return context.prisma.categories()

},

Queryproducts(root, args, context) {

return context.prisma.products()

},

Queryinventories(root, args, context){

return context.prisma.inventories()

},

Queryreorders(root, args, context){

return context.prisma.reorders()

},

productsByCategory(root, args, context){

return context.prisma.products({

where: {

category: {id: args.id}

},

})

},

inventoryByProducts(root, args, context){

return context.prisma.inventories({

where: {

product: {id: args.id}

},

})

},

},

3.

Product: {

category(root, args, context) {

return context.prisma.product({

id: root.id

}).category()

}

},

Inventory: {

product(root, args, context) {

return context.prisma.inventory({

id: root.id

}).product()

}

},

}

const server = new GraphQLServer({

typeDefs: './schema.graphql',

resolvers,

context: {

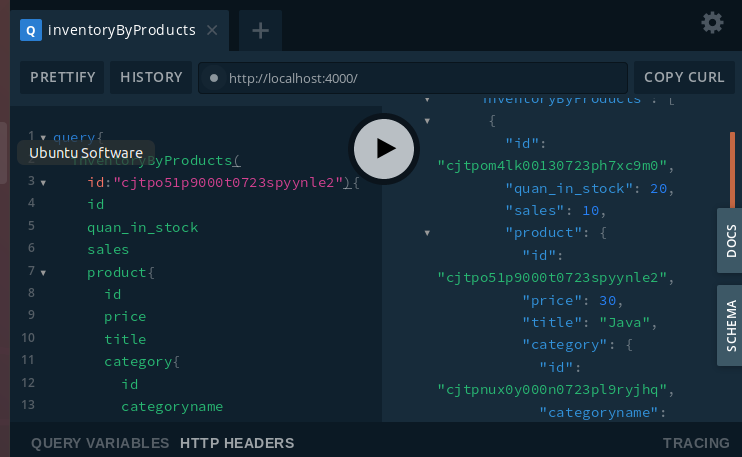
prisma

},

})

server.start(() => console.log('Server is running on http://localhost:4000'))

Nested query



4.

Mutation: {

createCategory(root, args, context) {

return context.prisma.createCategory(

{

categoryname: args.categoryname

},

)

},

createProduct(root, args, context) {

return context.prisma.createProduct(

{

title: args.title,

category: {

connect: {

id: args.category

}

},

actor: args.actor,

price: args.price,

special: args.special,

common\_prod\_id: args.common\_prod\_id

},

)

},

createInventory(root, args, context) {

return context.prisma.createInventory(

{

product: {

connect: { id: args.product }

},

quan\_in\_stock: args.quan\_in\_stock,

sales: args.sales,

},

)

},

createReorder(root, args, context) {

return context.prisma.createReorder(

{

prod\_id: {

connect: { id: args.prod\_id }

},

date\_low: args.date\_low,

quan\_low: args.quan\_low,

date\_reordered: args.date\_reordered,

quan\_reordered: args.quan\_reordered,

date\_expected: args.date\_expected

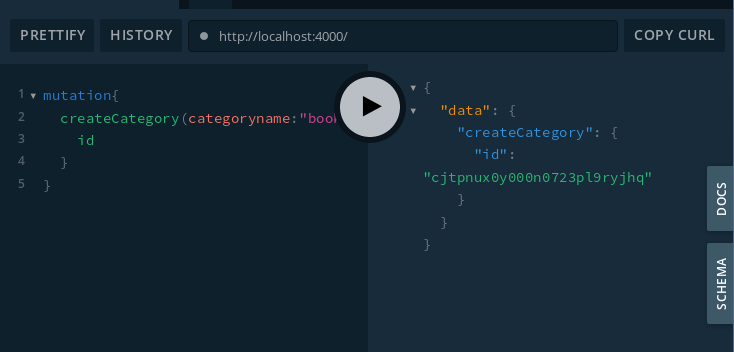
},

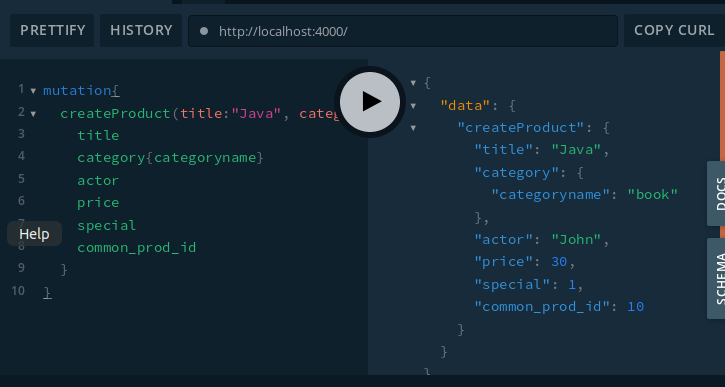
)

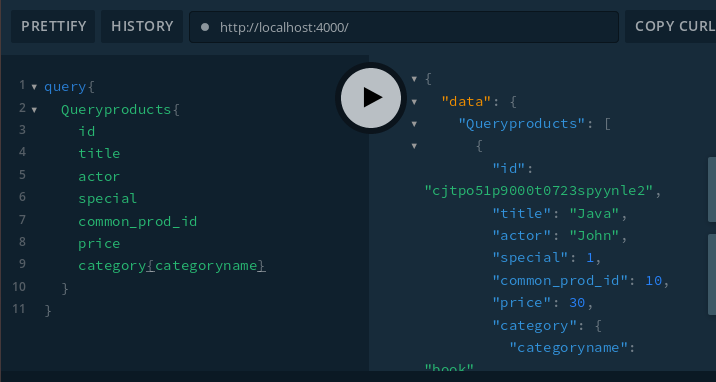
},

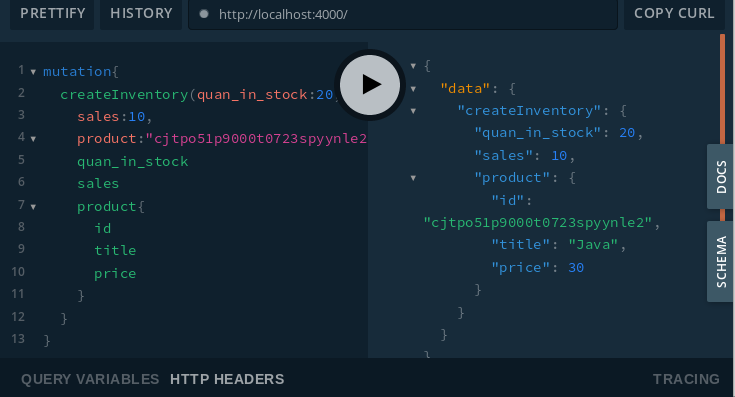
},

插入数据 add database

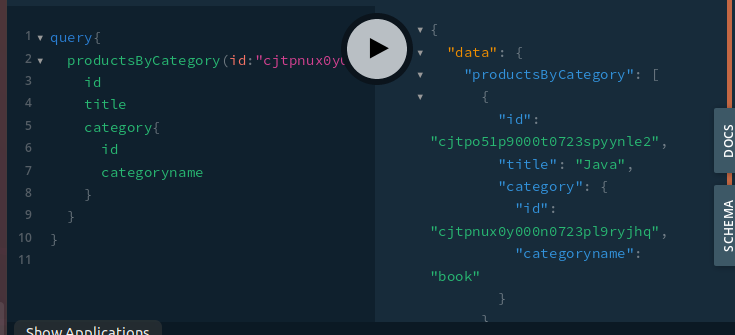


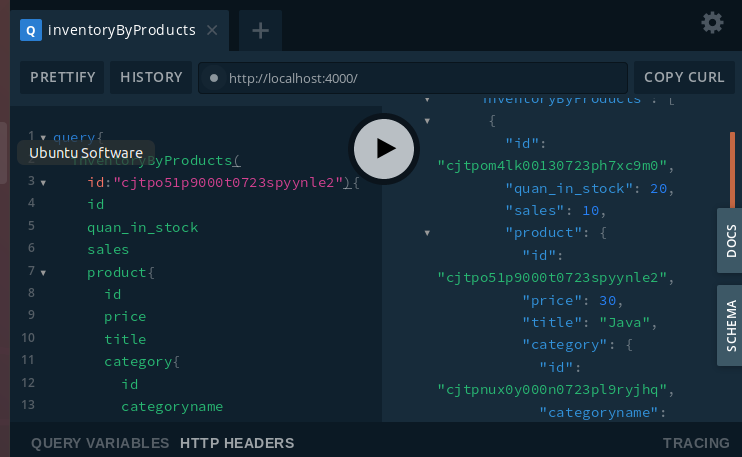






通过分类表查产品





5.

以上有运行结果