|  |  |  |
| --- | --- | --- |
| **CUSTOM TYPE** | ROOT QUERY TYPE | MUTATION |
| const <customTypeName> = new GraphQLObjectType ({  **name**: <General Name (Can be anything)>,  **description**: <General Description (Can be anything)>,  **fields**: **() => ({**  **field1**: {**type**: <**fieldType**>},  **field2**: {**type**: new **GraphQLList** <anotherCustomTypeName>,  **resolve(**parent, args**)**  **{**  **r**eturn \_.filter(<dataArray>, **{dataArray.id : parent.id}**)  **}**  }  **})**  })  **Note:**   * Here parent.id means < customTypeName >. id | const RootQueryType = new GraphQLObjectType ({  **name**:”RootQueryType”,  **description**: “<any description>”,  **fields:** {  **field1**: {  **type: <customType1>,**  **args: {<QueryDataNode>: {type: <** fieldType **>}},**  **resolve(parent,agrs) {**  return \_.find(<dataArray>, {<**QueryDataNode>**: **args.id**})  **}**  },  **field2**: {  **type**: new **GraphQLList**(**customType2**),  **args**: {},  **resolve** (parent, args) {  return <dataArray>  }  }  }  })  **Note:**   * **<QueryDataNode> is nothing but, dataNode in the dataArray, on which, we might be wanting to apply the condition** * **Field 1🡪 example for returning data based on condition applied** * **Field 2 🡪 example for returning entire data** | const Mutation = new GraphQLObjectType ({  **name**:”Mutation”,  **description**: “<any description>”,  **fields:** {  **field1**: {  **type: <customType1>,**  **args: {**  **<AddingDataNode1>: {type: <** fieldType **>},**  **<AddingDataNode2>: {type: <** fieldType **>}**  **},**  **resolve(parent,agrs) {**  let tempData = {  **AddingDataNode1**: args. **AddingDataNode1**,  **AddingDataNode2**: args. **AddingDataNode2**  }  return tempData;  **}**  },  **field2**: {  **<SAME AS ABOVE>**  }  }  }) |