To achieve full CRUD functionality on the Song model using Django Rest and Graphene-Django, you will need to define mutations for each CRUD operation (create, read, update, delete). Here's an example implementation:

1. Create a new song:

scss

Copy code

```
class CreateSongMutation(graphene.Mutation):
    class Arguments:
        title = graphene.String(required=True)
        artist_id = graphene.ID(required=True)
        album_id = graphene.ID(required=True)
        track_number = graphene.Int(required=True)
        length = graphene.String(required=True)

    song = graphene.Field(SongType)

    def mutate(self, info, title, artist_id, album_id, track_number, length):
        artist = Artist.objects.get(id=artist_id)
        album = Album.objects.get(id=album_id)
        song = Song.objects.create(title=title, artist=artist, album=album, track_number=track_number, length=length)
        return CreateSongMutation(song=song)
```

2. Update an existing song:

python

Copy code

```
class UpdateSongMutation(graphene.Mutation):
    class Arguments:
        id = graphene.ID(required=True)
        title = graphene.String()
```

```
        artist_id = graphene.ID()
        album_id = graphene.ID()
        track_number = graphene.Int()
        length = graphene.String()

    song = graphene.Field(SongType)

    def mutate(self, info, id, title=None,
artist_id=None, album_id=None, track_number=None,
length=None):
        song = Song.objects.get(id=id)
        if title:
            song.title = title
        if artist_id:
            song.artist =
Artist.objects.get(id=artist_id)
        if album_id:
            song.album = Album.objects.get(id=album_id)
        if track_number:
            song.track_number = track_number
        if length:
            song.length = length
        song.save()
        return UpdateSongMutation(song=song)
```

3. Delete a song:

python

Copy code

```
class DeleteSongMutation(graphene.Mutation):
    class Arguments:
        id = graphene.ID(required=True)

    success = graphene.Boolean()

    def mutate(self, info, id):
```

```
        song = Song.objects.get(id=id)
        song.delete()
        return DeleteSongMutation(success=True)
```

You can then add these mutations to your `Mutation` class:

scss

Copy code

```
class Mutation(graphene.ObjectType):
    create_song = CreateSongMutation.Field()
    update_song = UpdateSongMutation.Field()
    delete_song = DeleteSongMutation.Field()
```