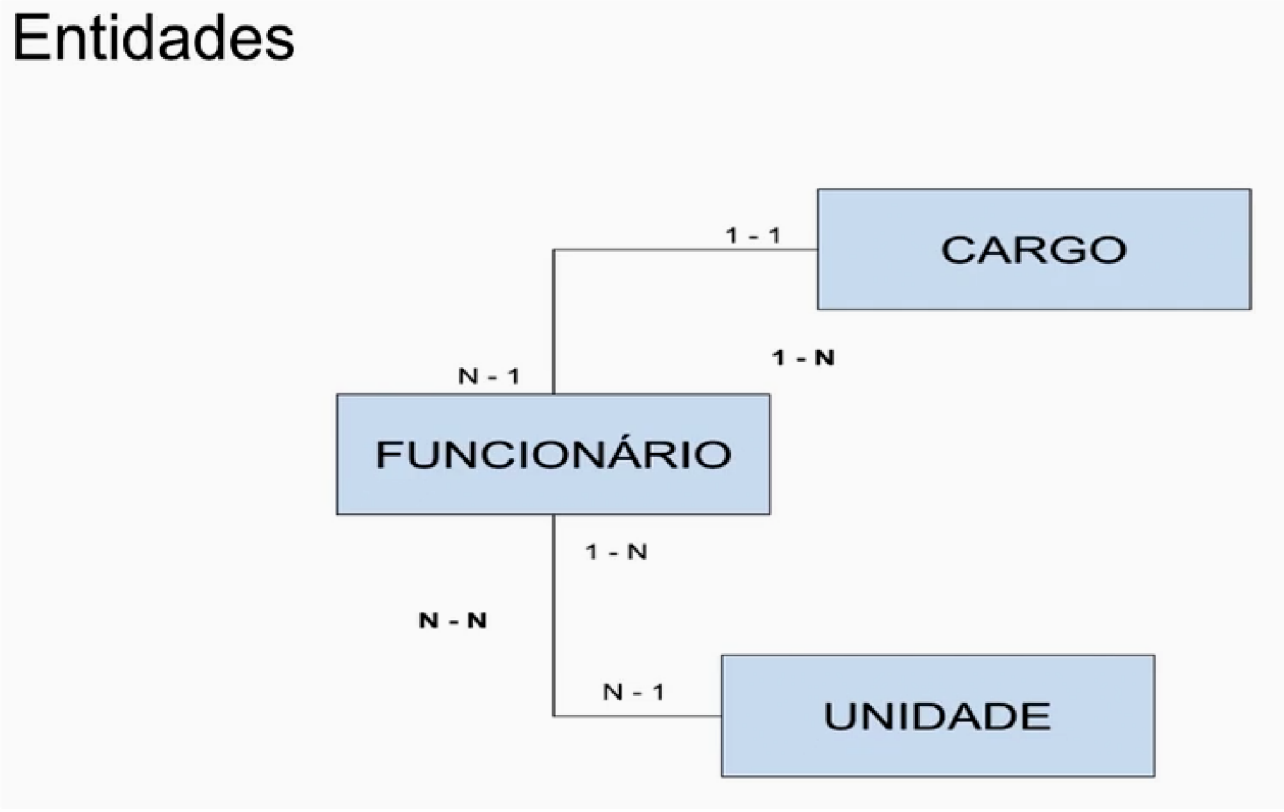
06

Gerando o banco de dados

Transcrição

[00:00] Olá. Bem-vindo a mais uma aula. Agora que já temos um sistema que se conecta a uma base de dados, podemos começar a desenvolver programas utilizando esse projeto. Sabendo disso, a empresa Recrutei entrou em contato com nós para que façamos um sistema de controle de funcionários para eles. O que se baseia esse sistema?



[00:20] Eles querem controlar os cargos de funcionários e quais as unidades um funcionário pode ou não adentrar e gerar alguns relatórios baseado nesses registros. Eles passaram para nós esse pequeno diagrama para que possamos observar como é a integração, o modelo lógico das entidades que eles desejam.

Então temos funcionário que se relaciona com cargo e temos funcionário que se relaciona com unidade.

[00:49] Então um funcionário pode até adentrar em unidades, uma unidade pode receber N funcionários, um funcionário pode ter um cargo e um cargo pode ter N funcionários. É mais ou menos essa a relação entre entidades que temos. O que vamos ver nessa aula? Nós temos entidades que precisamos representá-las tanto na nossa aplicação quanto também no nosso banco de dados.

[01:14] E vamos ver como criamos isso no Spring Data e criando essa entidade representada dentro da nossa aplicação Java, o próprio Spring Data já cria essa entidade também do banco de dados. Mas antes de começarmos a implementação, eu queria mostrar para vocês uma coisa. Esse aqui é a arquitetura do Spring Data. Então o que podemos ver?

[01:41] Dentro do Spring Data nós temos um Driver, temos um JDBC, um Hibernate e um JPA. Se caso você já tenha estudado, você sabe que esses componentes nós podemos encontrar eles apartados, ou seja, provavelmente já estudou algum componente desse fora desse ecossistema Spring Data. Mas qual é a parte legal do Spring Data? É que ele acopla todos esses componentes, todas essas coisas tudo dentro de um componente só.

[02:10] Então nós já temos todas essas ferramentas prontas para trabalhar. Por exemplo, se caso você já tenha trabalhado com JPA alguma vez, você sabe que para você conseguir executar alguns comandos em banco de dados, você vai precisa criar um `EntityManagerFactory`, pegar esse `EntityManagerFactory` para criar um `EntityManager` e aí você vai conseguir fazer ações no banco de dados.

[02:34] No caso aqui vai ser a mesma coisa, só que não vamos precisar implementar isso em código. O próprio Spring Data já implementa isso para nós. Então por isso que fica muito mais limpo, muito mais fácil utilizar o Spring Data na aplicação. Agora sem mais delongas, vamos codificar. Como tínhamos visto lá no modelo lógico, nós precisamos criar uma entidade.

[03:01] Então vou vir no pacote principal do Spring Data, dentro da classe Java, no pacote principal, vou clicar com botão direito "New > Package". O nome dessa Package eu vou dar de "orm". Apenas uma nomenclatura para que eu saiba que aqui estão as entidades de "orm". Entidades que o Spring Data gera para nós. Só a nomenclatura, caso você queira, você pode colocar o nome da entidade como você preferir.

[03:28] Vou clicar em "Finish", vou vir aqui dentro do pacote que eu acabei de criar, botão direito "New > Class". A entidade que eu vou criar é a de Cargo, a primeira entidade que está lá no modelo que a Recrutei passou para nós. Coloquei aqui o nome da classe de `Cargo` e vou clicar em "Finish". Pronto, já tenho a minha entidade `Cargo`.

[03:56] Agora que eu tenho uma entidade de `Cargo`, o que eu vou fazer? Primeiro eu vou colocar uma anotação do JPA do JPA Persistence que se chama `@Entity`. O que faz essa anotação? Nós finalizamos para Spring Data que essa classe é uma entidade. Nós vamos querer que ela crie uma entidade para nós a partir dessa classe. E eu também tenho outra anotação que é o `@Table`.

[04:23] Por que usamos esse `@Table`? O nome da entidade é `Cargo`, então se você criar só a quantidade ele vai criar como cargo no banco de dados. Só que o

`@Table` dá algumas propriedades para você mudar alguns comportamentos da tabela, como, por exemplo, o *name*. Eu posso vir aqui e falar, "Não, eu não quero que a minha entidade se chame 'cargo'. Eu quero que cidade se chame 'cargos', no plural".

[04:54] Então por isso que nós podemos utilizar essa notação também. A minha entidade aqui vai ter dois atributos. São eles o ID Controlador para que eu possa identificar um elemento através de um ID. Então vou colocar um `private Integer id` e ele vai ter também uma *String* que vai ser a descrição do cargo. Por exemplo, RH, desenvolvedor e por aí vai.

[05:34] Qualquer cargo, qualquer função que tenha dentro da empresa ele vai colocar essa descrição aqui dentro. Então `private String descricao`, agora eu também vou colocar duas anotações aqui. A primeira é o `@Id` que também é do Java Persistance. Por que eu tenho que colocar o `@Id`? Para sinalizar o Spring Data que esse atributo aqui é o meu atributo ID que é o atributo que eu vou controlar a minha entidade.

[06:21] E também vou colocar um `@GeneratedValue` que é para ele gerar os valores automáticos para mim para que eu não precise ficar citando um, dois, três. O próprio Framework já vai fazer isso automaticamente para mim. E aqui vou colocar a estratégia de criação desse ID automático que eu quero, que no caso eu vou colocar o `@GenerationType.IDENTITY` que ele vai utilizar da forma "IDENTITY".

[06:55] Então vai criar os números sequenciados. Feito isso eu já tenho as duas entidades criadas. Vou vir aqui em cima na aba "Source" e vou vir em "Generate Getters and Setters". Vou selecionar os dois atributos e vou clicar em "Generate". Pronto, eu já tenho os "Getters and Setters" gerado da minha aplicação. Agora vou salvar e vou simplesmente executar a aplicação.

[07:29] Venho na classe principal, botão direito, vou clicar em "Run As > Java Application". Cliquei no "Java Application". Agora só esperamos a aplicação *startar*. Deixa eu maximizar para vermos quando ela finalizar de *startar*,

quando ela estiver em pé. Então vamos lá. Está quase. Pronto, ela *startou*. Já completou aqui, iniciado. Agora eu vou lá no DBeaver.

[08:05] Deixa eu trazer o DBeaver para cá. Já estou com ele conectado com o LocalHost, vou vir na pasta de banco de dados e vou clicar em "Renovar" ou você pode apertar F5 para que ele faça o *Refresh*. Agora no meu banco de dados da Alura, se eu vir na pasta "tables", eu já tenho a minha tabela "Cargo" pronta para uso. Deixa ele carregar. Aqui, "id" e "descricao".

[08:40] Então veja, com muito pouco, só criando uma entidade, só colocando algumas anotações, o Spring Data já gera essa tabela também no nosso banco de dados, por isso que traz uma facilidade gigante utilizar esse Framework. Espero que vocês tenham gostado dessa aula. Vejo vocês na próxima.