roteirinho lp1

dependencies- pesquisar h2 maven e hibernate maven
(maven repository)

apagar ultima linha caso precisar

criar arquivo hibernate.cfg dentro dos recourses

criar classe hibernate util e Main dentro do pacote util dentro do java

criar classes

import jakarta.persistence.*; !!!!!

@entity e @table em todas as classes

criar o pacote entity em java em tds que tem @entity

atributos

para ids auto incremento - @GeneratedValue(strategy= GenerationType.IDENTITY)

@Column dps do generated

hibernate cfg tem os mappings
    <mapping class="entity.NomeDaClasse"/>

import java.util*; !!!!!!!!!

para lista— private List<Aluno> alunos;

cascade salva tudo de uma vez

to string para buscar tudo no bdd e exibir todas as informações no console

relacionamentos:
sempre olhar a partir da classe que está
exemplo: se caso o nome da classe for viagem você precisa se perguntar quantas viagens para um piloto!!
nesse caso, ManyToOne

cascade garante que quando se faça o cadastro de uma viagem, também se faça de um piloto.

@ManyToOne(cascade = CascadeType.ALL)

JoinColumn é uma coluna de junção, uma coluna que possui a chave estrangeira da outra classe (sem ser a que tem o JoinColumn)
colocar o nome como tabelaRelacionada_id
@JoinColumn(name = “piloto_id”)

ManyToMany cria uma tabela de junção (ja que em bdd, muitos pra muitos cria uma tabela nova). Depois do nome, colocar virgula e colocar a primeira coluna de junção, que é a da tabela em que voce esta!!!, depois a outra tabela relacionada.
colocar o nome como tabela1-tabela2
@JoinTable(name = “viagem-passageiro”,
joinColumns = @JoinColumn(name = “classeatual_id”), inverseJoinColumns = @JoinColumn(name = “classerelacionada_id”))

o construtor das classes q tem relacionamento só pode envolver atributos q não estão relacionando

na main, se adiciona o atributo com o set
e o nome do objeto que se vai passar (provavelmente uma variável ja criada)
viagem.setPiloto(piloto);
viagem.setPassageiros(List.of(passageiro1, passageiro2));

-para exibir, gerar um toString

-para os ids de autoincremento, abrir uma sessão :

-Session session = HibernateUtil.getSessionFactory().openSession();

para salvar os elementos no bdd, abrir uma transição e uma persistência :

-Transaction transaction = session.beginTransaction();

session.persist(classe);

para transmitir todos os elementos da tabela:
List <Turma> turmas = session.createQuery(“from tabela”, Turma.class).list();

para percorrer, fori e colocar o incremento pelo tamanho do vetor (objeto.size()) e sout (objeto.get(i));

* import sempre precisa ser do hibernate!!

*listas precisam ser inicializadas colocando o atributo da lista no construtor (this.atb = new ArrayList<>();