



TRABALHO INTEGRADO – PW e BD

Proposta

Este projeto faz parte da disciplina de Programação Web II em integração com a disciplina de Banco de Dados.

A proposta é trabalhar os conceitos de modelagem relacional e mapeamento objeto-relacional (JPA/Hibernate) a partir de um estudo de caso baseado em um aplicativo de restaurantes estilo iFood.

Cada grupo ficará responsável por um conjunto de entidades do modelo (bounded context), devendo:

- 1. Elaborar o script SQL de criação das tabelas correspondentes;*
- 2. Implementar as entidades JPA com anotações de mapeamento, refletindo a modelagem definida.*

Objetivos da Atividade

- Compreender a transição entre modelo de classes UML → modelo relacional SQL → entidades JPA.*
- Aplicar conceitos de chaves primárias, chaves estrangeiras e constraints.*
- Desenvolver mapeamentos corretos de relacionamentos (1:1, 1:N, N:N).*
- Trabalhar em equipe com controle de versão (Git/GitHub), realizando entregas via Pull Request (PR).*



TRABALHO INTEGRADO – PW e BD

Estrutura de equipes

Dividiremos o desenvolvimento do projeto entre 7 equipes, sendo 5 equipes com 5 integrantes e 2 equipes com 6 integrantes. Cada equipe ficará responsável por um conjunto de entidades do sistema, implementando o script SQL e o mapeamento JPA, além de garantir a consistência entre os dois.

A equipe fará um FORK no repositório <https://github.com/emarion17/etec-food.git>

Grupo 1 – Cadastro e Gerenciamento de Usuários (5 alunos)

Entidades:

- Usuario
- Permissao
- StatusUsuarioEnum

Integrantes: Enzo Correa, Gustavo Rocha, Heitor, Marina e Julia

Banco de dados:

Tabela Usuário não aplicou o CHECK no campo status, não tem nenhuma regra (Not null) Não criou a tabela Permissão, mas fez tabelas que não eram do seu grupo (Restaurante, Horário Funcionamento).

Classes: Não subiu para o git nenhuma classe



TRABALHO INTEGRADO – PW e BD

Subiu as classes no dia 29-09. O atributo id na classe Permissão está for a da convenção.

Anotação @ManyToMany errado:
@ManyToMany(fetch = FetchType.LAZY)
@JoinTable(name = "")
private List<Permissao> permissaoList;

Anotação @Column(name = "tentativaAcesso") errada, no banco sua coluna chama tentativasAcesso. O campo status no banco está com outro nome, portanto a anotação @Column é necessária.

Grupo 2 – Cadastro e Gerenciamento de Restaurantes (6 alunos)

Entidades:

- *Restaurante*
- *HorarioFuncionamento*
- *TipoCozinhaEnum*
- *RestauranteFormaPagamento*
- *RestauranteFormaPagamentold*

Integrantes: Matheus Rocha, Kaio, Sarah, Bruno, Samuel e Larissa

Tabela Restaurante no campo tipoCozinha que é uma enum deveria ter colocado

a constraint check: CONSTRAINT CK_Restaurante_TipoCozinha CHECK
(tipoDeCozinha IN ('CHINESA', 'JAPONESA', 'MEXICANA',
'MINEIRA', 'BAIANA', 'LANCHES', 'HAMBURGER', 'ARABE', 'ITALIANA', 'VARIADA'
))

Tabela HorarioFuncionamento não usou nenhuma restrição.



TRABALHO INTEGRADO – PW e BD

Classe Restaurante: No banco a tabela chama Restaurante e vc colocou na anotação TBL_Restauranta. Id no banco não está setado como identity, portanto não existe a anotação @GeneratedValue, eu coloquei na versão integrada, e na classe só tem o campo id?

Não está preenchido os demais campos.

Classe HorárioFuncionamento: diaSemana no banco está varchar na classe DayOfWeek.

Classe RestauranteFormaPagamento ainda não explicamos a anotação @EmbeddedId

Grupo 3 – Pagamento (6 alunos)

Entidades:

- FormaPagamento
- Pagamento
- TipoFormaPagamentoEnum
- StatusPagamentoEnum

Integrantes: Daphne, Felipe Cardozo, Guilherme, Kamila, Kauã, Kevin

Correção do Banco de dados: Não colocou a constraint check nas enumerações tipo e status:

```
CONSTRAINT CK_FormaPagamento_Tipo CHECK (TP_TIPO IN ('CARTAO_CREDITO',  
'CARTAO_DEBITO', 'VALE_REFEICAO'))
```

Correção do Java: Forma de pagamento não fez a anotação @Enumerated, o mesmo acontece na classe Pagamento.

Faltou a anotação @ManyToMany



TRABALHO INTEGRADO – PW e BD

Rever os mapeamentos:

@OneToMany

@JoinColumn(name = "ID_FORMA_PAGAMENTO")

private FormaPagamento formaPagamento;

@OneToOne

@JoinColumn(name = "ID_PEDIDO")

private Pedido pedido;

Grupo 4 – Cardápio e Itens (5 alunos)

Entidades:

- Cardapio
- ItemCardapio
- TipoItemCardapioEnum

Integrantes: Luighi, Gustavo Alberico, Miguel, Giulia, Bianca

Correção do Banco de dados: Não encontrei o banco de dados

Entregou dia 29-09: Na tabela cardapio deveria ter apenas o id e um alter table para adicionar restaurante, colocou o campo, mas não fez a definição da FK.

Na tabela Item_cardapio: Nome da tabela é ITEM_CARDAPIOO? Tipo não precisa de 100 caracteres, e faltou a constraint check.

Correção do Java: Não encontrei a enumeração TipoCardapio Enum, não encontrei a classe Cardapio e a classe ItemCardapio só tem o id, onde estão os demais campos?



TRABALHO INTEGRADO – PW e BD

Grupo 5 – Cliente (5 alunos)

Entidades:

- Cliente
- Entrega

Integrantes: João vitor Andrade, Victor Porfirio, Enzo Catarino, Enzo Holanda, Gabriel Basilio

Correção do banco de dados: Colocou o Cpf como chave primária, deixou uma vírgula no último campo de cliente, poderia ter usado mais constraint como unique para email. Os tamanhos também poderiam ter sido melhor avaliado, cep varchar(100)?

Correção do Java: Tabela Cliente e Entrega não tem a anotação @Entity, Está com erro no código no relacionamento pedido.

Em cliente vcs colocaram @GeneratedValue que no banco não é e não teria sentido também.

Grupo 6 – Pedido (5 alunos)

Entidades:

- Pedido
- ItemPedido
- Avaliacao
- StatusPedidoEnum



TRABALHO INTEGRADO – PW e BD

Integrantes: João Vitor Acioli, Matheus Aurichio, Thiago, Felipe Santos, Gabriel Chaves

Correção do banco de dados: Não colocou restrição check no status e tipo veículo, item cardápio não era do seu grupo portanto deveria estar com alter table.

Correção do Java: Tabela Pedido não tem a anotação @Enumerated

Grupo 7 – Entregadores e Logística (5 alunos)

Entidades:

- Entregador
- Veículo
- TipoVeiculoEnum

Integrantes: João Pedro, Matheus Guerino, Nikolas, Ricardo, Vitor Mathueus,

Correção do banco de dados: Não colocou restrição check.

Correção do Java: Fez todas as anotações

Entregáveis

Cada equipe deve entregar:

1. Script SQL

- Criação das tabelas correspondentes ao seu grupo.
- Definição de **chaves primárias, estrangeiras e constraints** (NOT NULL, UNIQUE, CHECK, etc.).
- Tipos de dados adequados (VARCHAR, DATE, BOOLEAN, DECIMAL, etc.).

2. Entidades JPA



TRABALHO INTEGRADO – PW e BD

- *Uso correto das anotações (@Entity, @Id, @GeneratedValue, @Enumerated, @OneToMany, @ManyToOne, @ManyToMany, @EmbeddedId, etc.).*
- *Enum mapeado como @Enumerated(EnumType.STRING).*
- *Relacionamentos configurados conforme o modelo UML.*

3. Pull Request (PR) no GitHub

- *Mensagem de commit clara e objetiva.*
- *PR com descrição do que foi feito.*
- *Organização do código (pacotes, indentação, nomes consistentes).*

Critérios de Avaliação

A nota será atribuída considerando:

1. Modelagem SQL (40%)

- *Correção na criação das tabelas.*
- *Uso adequado de chaves primárias e estrangeiras.*
- *Consistência com o modelo UML.*
- *Uso de constraints apropriadas.*

2. Mapeamento JPA (40%)

- *Implementação correta das entidades com anotações.*
- *Enum mapeado corretamente.*
- *Relacionamentos entre entidades de acordo com o diagrama.*
- *Consistência entre o SQL criado e o schema gerado pelo JPA.*



TRABALHO INTEGRADO – PW e BD

3. Qualidade da Entrega (20%)

- Organização do código e estrutura de pacotes.
- Qualidade das mensagens de commit e descrição do PR.
- Clareza e consistência na implementação.

Forma de Entrega

- O professor disponibilizará o repositório inicial no GitHub.
- Cada grupo fará o clone do repositório, implementará suas entidades e scripts, e abrirá um Pull Request (PR).
- O PR será utilizado como marco da entrega e ponto de avaliação.