Faculdade de Engenharia da Universidade do Porto

Empresa de Transporte de Bens

Base de Dados

2ª Entrega – Grupo 1005

2021/2022 – Licenciatura em Engenharia Informática e Computação

Dinis Ribeiro dos Santos Bessa de Sousa

(up202006303@edu.fe.up.pt)

Francisca Oliveira e Silva

(up202005140@edu.fe.up.pt)

Maria Eduarda Pacheco Mendes Araújo

(up202004473@edu.fc.up.pt)

Docentes:

Carla Alexandra Teixeira Lopes (ctl@fe.up.pt)

Michel Celestino Paiva Ferreira (mpferrei@fc.up.pt)

Pedro Emanuel Cardoso de Sousa (pesousa@fe.up.pt)

Introdução

Relatório elaborado no âmbito da Unidade Curricular: Base de Dados lecionada no 2º ano do ciclo de estudos de Licenciatura em Engenharia Informática e Computação na FEUP (Faculdade de Engenharia da Universidade do Porto).

Ao longo deste projeto será elaborada uma base de dados capaz de implementar o modelo de negócio de uma empresa de entrega de bens.

No relatório consta o modelo UML para a base de dados deste negócio, bem como a indicação das diferentes classes e atributos.

Índice

Introdução 2
Contexto 3
Diagrama UML 4
Diagrama UML Revisto 5
Esquema Relacional 6
Análise de Dependências Funcionais 7
Restrições 9

Contexto

Base de dados para a gestão de encomendas e funcionários para entregas e transporte de produtos (ex. FedEx).

Os funcionários estão divididos entre condutor e funcionários do posto de entrega e, tal como os clientes, são pessoas. De todas as pessoas são guardados o nome, a data de nascimento, o telefone, o email e a morada.

De todos os funcionários é guardado o seu salário. Dos funcionários do posto é necessário saber quais os seus horários de trabalho. Dos condutores é guardada informação sobre quais as categorias de veículos que podem conduzir. Da mesma forma, de cada carro é guardada a sua categoria.

Dos clientes é necessário guardar o NIF e o NIB. Um cliente pode realizar várias encomendas. Destas é necessário saber qual o posto onde o cliente deixa a remessa e qual o posto final. É ainda importante guardar o peso e volume.

De modo a realizar o transporte, são definidas rotas entre os diferentes pontos de entrega. Um determinado carro executa uma dada rota recolhendo e entregando encomendas. Deste modo, ao longo de uma viagem um só carro é capaz de transportar várias encomendas ao longo de uma dada rota.

Uma rota tem um posto de entrega inicial e um posto de entrega final, e pode ter vários postos de entrega entre eles.

Diagrama UML

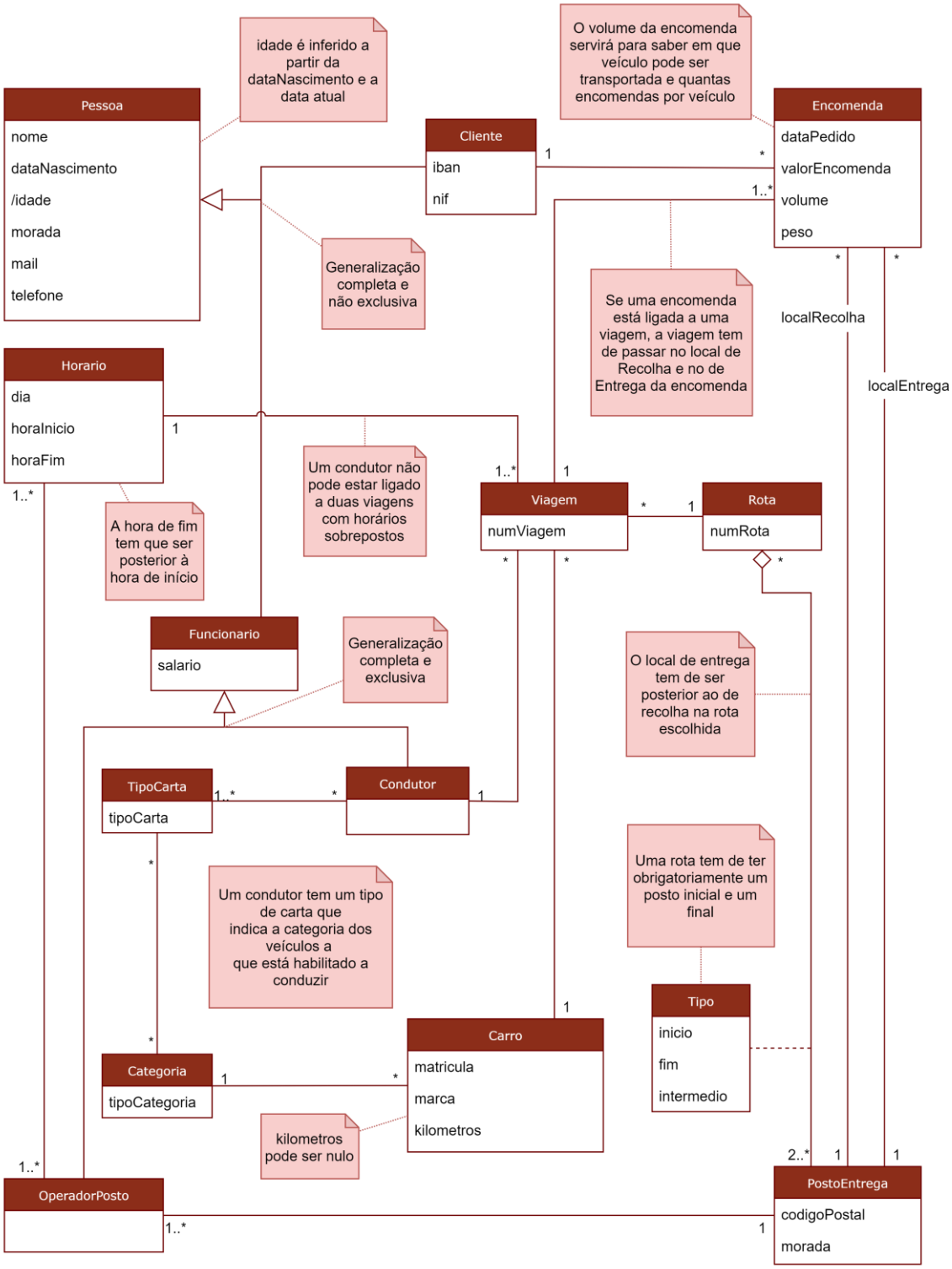
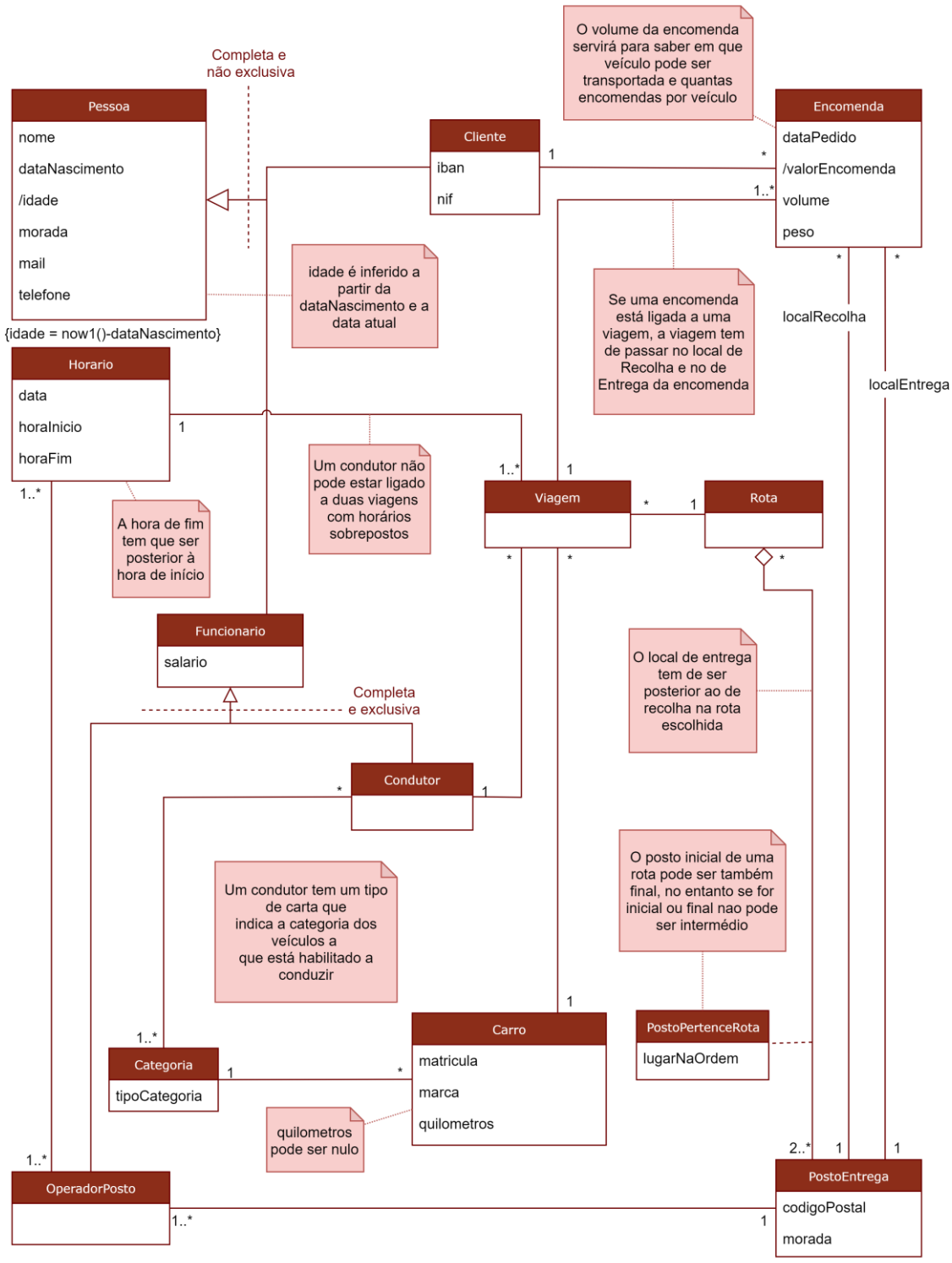


Diagrama UML Revisto



Esquema Relacional

Pessoa(idPessoa, nome, dataNascimento, morada, mail, telefone)

Cliente(idCliente->Pessoa, nib, nif)

PostoEntrega(idPostoEntrega, codigoPostal, morada)

Funcionario(idFuncionario->Pessoa, salario)

OperadorPosto(idOperadorPosto->Funcionario, idPostoEntrega->PostoEntrega)

Horario(idHorario, data, horaInicio, horaFim)

HorarioOperadorPosto(idOperadorPosto->OperadorPosto, idHorario->Horario)

Rota(idRota)

PostoPertenceRota(idRota->Rota, idPostoEntrega->PostoEntrega, lugarNaOrdem)

Condutor(idCondutor->Funcionario)

Categoria(idCategoria, tipoCategoria)

PodeConduzir(idCondutor->Condutor, idCategoria->Categoria)

Carro(idCarro, matricula, marca, quilometros, idCategoria->Categoria)

Viagem(idViagem, idRota->Rota, idCarro->Carro, idCondutor->Condutor, idHorario->Horario)

Encomenda(idEncomenda, dataPedido, volume, peso, idPostoEntrega->PostoEntrega, idPostoRecolha->PostoEntrega, idViagem->Viagem, idCliente->Cliente)

Análise de Dependências Funcionais

Pessoa(idPessoa, nome, dataNascimento, morada, mail, telefone)

idPessoa → nome, dataNascimento, morada, mail, telefone

Cliente(idCliente->Pessoa, nib, nif)

idPessoa → nib, nif

PostoEntrega(idPostoEntrega, codigoPostal, morada)

idPostoEntrega → codigoPostal, morada

Funcionario(idFuncionario->Pessoa, salario)

idFuncionario → salario

OperadorPosto(idOperadorPosto->Funcionario, idPostoEntrega->PostoEntrega)

idOperadorPosto → idPostoEntrega

Horario(idHorario, data, horaInicio, horaFim)

idHorario → data, horaInicio, horaFim

HorarioOperadorPosto(idOperadorPosto->OperadorPosto, idHorario->Horario)

Apenas contém dependências funcionais triviais.

Rota(idRota)

Apenas contém dependências funcionais triviais.

PostoPertenceRota(idRota->Rota, idPostoEntrega->PostoEntrega, lugarNaOrdem)

Apenas contém dependências funcionais triviais.

Condutor(idCondutor->Funcionario)

Apenas contém dependências funcionais triviais.

Categoria(idCategoria, tipoCategoria)

idCategoria → tipoCategoria

PodeConduzir(idCondutor->Condutor, idCategoria->Categoria)

Apenas contém dependências funcionais triviais.

Carro(idCarro, matricula, marca, quilometros, tipoCategoria->Categoria)

idCarro → matricula, marca, quilometros, tipoCategoria

Viagem(idViagem, idRota->Rota, idCarro->Carro, idCondutor->Condutor, idHorario->Horario)

idViagem → idRota, idCarro, idCondutor, idHorario

Encomenda(idEncomenda, dataPedido, volume, peso, idPostoEntrega->PostoEntrega, idPostoRecolha->PostoEntrega, idViagem->Viagem, idCliente->Cliente)

> idEncomenda → dataPedido, volume, peso, idPostoEntrega, idPostoRecolha, idViagem, idCliente

Podemos, assim, concluir que todas as relações da base de dados seguem tanto a Forma Normal Boyce-Codd (BCNF) como a terceira forma normal (3NF).

BCNF: Dada uma qualquer relação, esta cumpre com a BCNF se para qualquer dependência não trivial $\bar{A} \rightarrow B$, $\bar{A}$ é uma (super)chave da relação.

3NF: Dada uma qualquer relação, esta cumpre com a 3NF se para qualquer dependência não trivial $\bar{A} \rightarrow B$, $\bar{A}$ é uma (super)chave da relação OU B é constituído apenas por atributos primos.

Para qualquer relação da base de dados, em todas as suas Dependências Funcionais (FDs), a partir da parte esquerda da FD ($\bar{A}$) é possível descobrir todos os atributos da relação, o que significa que $\bar{A}$ é uma (super)chave. Isto é uma condição suficiente para puder afirmar que todas as relações cumprem com a BCNF e com a 3NF.

Restrições

Pessoa

- Não pode haver duas pessoas com o meso ID:
    - idPessoa PRIMARY KEY
- Todas as pessoas devem ter um nome, data de nascimento, morada, mail e telefone
    - nome NOT NULL
    - dataNascimento NOT NULL
    - morada NOT NULL
    - mail NOT NULL
    - telefone NOT NULL
- O número de telemóvel não poderá ser negativo.
    - CHECK (telefone > 0)
- A data de nascimento não poderá ser inferior à data atual;

Cliente

- Não pode haver dois clientes com o mesmo ID, sendo que o ID do cliente corresponde ao ID de pessoa na tabela Pessoa:
    - idCliente PRIMARY KEY REFERENCES Pessoa(idPessoa)
- Todos os clientes devem ter um NIF e um IBAN associados:
    - nib NOT NULL
    - nif NOT NULL

PostoEntrega

- Não pode haver dois postos de entrega com o meso ID:
    - idPostoEntrega PRIMARY KEY
- Todos os postos de entrega devem ter um código postal e uma morada.
    - codigoPostal NOT NULL
    - morada NOT NULL

Funcionário

- Não pode haver dois funcionários com o mesmo ID, sendo que o ID do funcionário corresponde ao ID de pessoa na tabela Pessoa:
    - idFuncionario PRIMARY KEY REFERENCES Pessoa(idPessoa)
- Todos os clientes devem ter um salário associado:
    - salario NOT NULL

OperadorPosto

- Não pode haver dois operadores de posto com o mesmo ID, sendo que o ID do operador de posto corresponde ao ID de funcionário na tabela Funcionario:
    - idOperadorPosto PRIMARY KEY REFERENCES Funcionario(idFuncionario)
- POSTO ENTREGA
- O ID de posto de entrega de OperadorPosto corresponde a um ID de posto na tabela PostoEntrega:
    - idPostoEntrega REFERENCES PostoEntrega(idPostoEntrega)

Horario

- Não pode haver dois horários com o meso ID:
    - idHorario PRIMARY KEY
- Todos os horários devem ter uma data, hora de início e uma hora de fim.
    - data NOT NULL
    - horaInicio NOT NULL
    - horaFim NOT NULL
- A hora de fim terá de ser posterior à hora de início
    - CHECK (horaFim > horaInicio)
- Para um dado dia, não existem horários repetidos
    - UNIQUE (data, horaInicio, horaFim)

HorarioOperadorPosto

- Nesta relação um Operador de Posto trabalha num Horário, sendo que cada par operador horário é único.
    - idPessoa REFERENCES Funcionario(idPessoa)
    - idHorario REFERENCES Horario(idHorario)
    - PRIMARY KEY(idPessoa, idHorario)

Rota

- Não pode haver duas rotas com o mesmo ID:
    - idRota PRIMARY KEY

PostoPertenceRota

- Cada posto pertence a uma rota, sendo a rota e o posto de entrega referenciados por chave estrangeira.
    - idRota REFERENCES Rota(idRota)
    - idPostoEntrega REFERENCES PostoEntrega(idPostoEntrega)
- Cada combinação de rota, posto de entrega e lugarNaOrdem de posto será única.
    - PRIMARY KEY(idPostoEntrega, idRota, lugarNaOrdem)

Condutor

- Não pode haver dois condutores com o mesmo ID, sendo que o ID do condutor corresponde ao ID de funcionario na tabela Funcionário:
    - idCondutor PRIMARY KEY REFERENCES Funcionário(idFuncionario)

Categoria

- Não pode haver duas categorias com o meso ID:
    - idCategoria PRIMARY KEY
- Todas as categorias têm um atributo que indica a categoria.
    - tipoCategoria NOT NULL
- Não existem duas categorias com o mesmo nome.
    - tipoCategoria UNIQUE

PodeConduzir

- Nesta relação um Condutor pode conduzir uma Categoria de carros, sendo que cada par condutor categoria é único.
    - idCondutor REFERENCES Condutor(idCondutor)
    - idCategoria REFERENCES Categoria(idCategoria)
    - PRIMARY KEY(idCondutor, idCategoria)

Carro

- Não pode haver dois carros com o meso ID:
    - idCarro PRIMARY KEY
- Todos os carros devem ter matrícula, marca e número de quilómetros percorridos.
    - matricula NOT NULL
    - marca NOT NULL
    - quilometros NOT NUL
- Cada matrícula deve ser única para cada carro:
    - matricula UNIQUE
- O ID de categoria do carro corresponde a um ID de categoria na tabela Categoria:
    - idCategoria REFERENCES Categoria(idCategoria)

Viagem

- Não pode haver duas viagens com o mesmo ID:
    - idViagem PRIMARY KEY
- Os restantes atributos de viagem serão chaves estrangeiras para as relações Rota, Carro, Condutor e Horario
    - idRota REFERENCES Rota(idRota)
    - IdCarro REFERENCES Carro(idCarro)
    - IdCondutor REFERENCES Condutor(idCondutor)
    - IdHorario REFERENCES Horario(idHorario)

Encomenda

- Não pode haver duas encomendas com o meso ID:
    - idEncomenda PRIMARY KEY
- Todas as encomendas devem ter uma data de pedido, um volume e um peso.
    - dataPedido NOT NULL
    - volume NOT NULL
    - peso NOT NULL
- O ID do posto onde a encomenda vai ser entregue e o ID do posto onde a encomenda vai ser recolhida são IDs da tabela PostoEntrega. Terá também um ID de viagem, que corresponde a um ID da tabela Viagem e a um ID de cliente, retirado da tabela Clientes.
    - idPostoEntrega REFERENCES PostoEntrega(idPostoEntrega)
    - idPostoRecolha REFERENCES PostoEntrega(idPostoEntrega)
    - idViagem REFERENCES Viagem(idViagem)
    - idCliente REFERENCES Cliente(idCliente)
- O posto de entrega tem de ser diferente do posto de recolha.
- CHECK (idPostoEntrega <> idPostoRecolha)

Nota: As restrições presentes no UML e ainda outras que consideramos que poderão ser importantes (tais como: o mesmo condutor/carro não puder efetuar duas viagens ao mesmo tempo, o posto de entrega e recolha de uma encomenda devem ter uma ligação (na tabela PostoPertenceRota) à rota percorrida pela viagem, a data da encomenda ter de ser anterior ao início da viagem, estabelecer idade mínima para trabalhador, etc...) que não estão descritas acima, nas restrições, não foram ainda implementadas por dependerem de atributos de diferentes classes, que serão possíveis de fazer cumprir mais tarde no trabalho.