Faculdade de Engenharia da Universidade do Porto

Centro Comercial (CC)

Contexto e UML – *Entrega 3*

Bases de Dados 2019/2020 – MIEIC

24/05/2020

Turma 3

Grupo 306:

Ana Teresa Feliciano da Cruz up201806460@fe.up.pt

André Filipe Meireles do Nascimento up201806461@fe.up.pt

Pedro Miguel Pires Coelho up201806802@fe.up.pt

Contexto

Introdução

Com o objetivo de melhorar a gestão do Centro Comercial alterámos ligeiramente alguns atributos de classes e ainda, acrescentámos e removemos algumas. Desta forma, temos em conta mais aspetos ligados aos funcionários e ao funcionamento das lojas. Segue-se uma explicação das classes e das ligações entre elas.

- ### Centro Comercial (CC)

O Centro Comercial, de nome CC, está associado a uma Localização, com os atributos morada, localidade e código postal, e está dividido em pisos representados pela classe Piso identificado pelo número. A cada um destes está associado um conjunto de lojas, seguranças e empregados de limpeza.

- ### Loja

A classe Loja representa todas as lojas do CC. Cada loja tem um nome e um id(número) único que a identifica e está associada a uma Categoria que representa o tipo de loja, ou seja, se é de calçado, vestuário, bijutaria, etc. Cada loja pode ter mais do que uma categoria, uma vez que a mesma loja pode vender tanto roupa como calçado, por exemplo.

- ### Produto

Os produtos estão associados à classe Loja. A cada loja está associado um conjunto de produtos identificados pelo nome, código definido pela loja, preço disponível ao público e quantidade do produto na respetiva loja. A cada produto está também associado um tipo que explicita o tipo de produto que é, por exemplo, fato ou botas.

- ### Cliente

Tanto a classe Cliente como Funcionário são subclasses da classe Pessoa, de modo a evitar repetição de atributos comuns como nome, nif, data de nascimento, telefone e email. Um cliente quando faz uma compra, os produtos escolhidos, o dia e a hora da compra e um id (número identificativo da compra) são representados na classe Compra, onde também se associa a quantidade de cada produto comprado. É de realçar que a uma compra tem sempre pelo menos um produto associado.

- Funcionário

Esta classe representa todos os funcionários do CC. Cada funcionário tem um salário e um Horário associado. Esta classe divide-se também em três subclasses: Empregado de loja, Segurança e Empregado de Limpeza.

Cada Empregado de Loja está associado a uma loja, sendo um deles o gerente, ou seja, o responsável de loja. Deste modo, uma loja tem sempre, pelo menos, um empregado de loja.

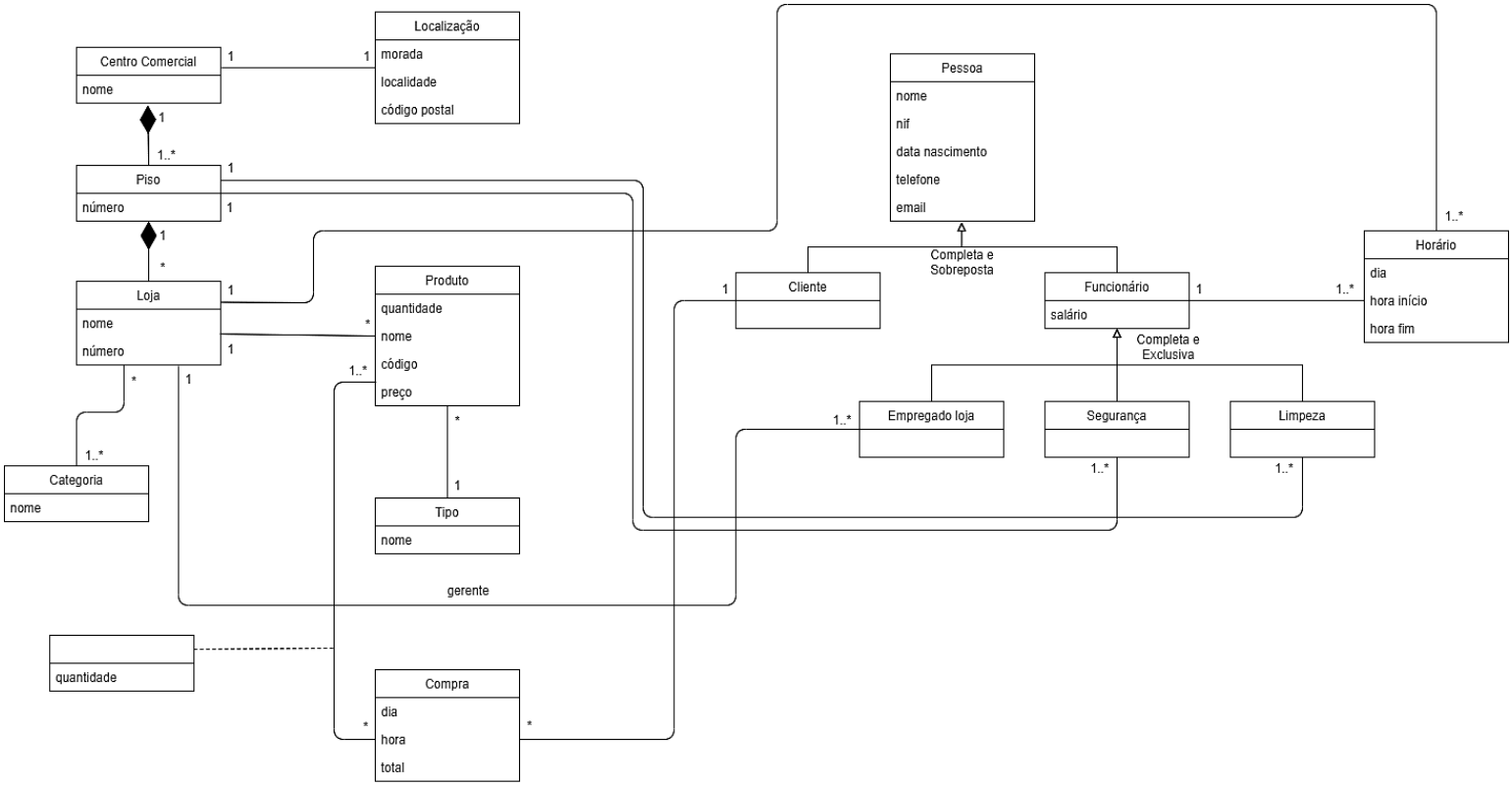
Os Seguranças e Empregados de limpeza estão associados a um só Piso. Cada piso tem pelo menos um segurança e empregado de limpeza.

- Horário

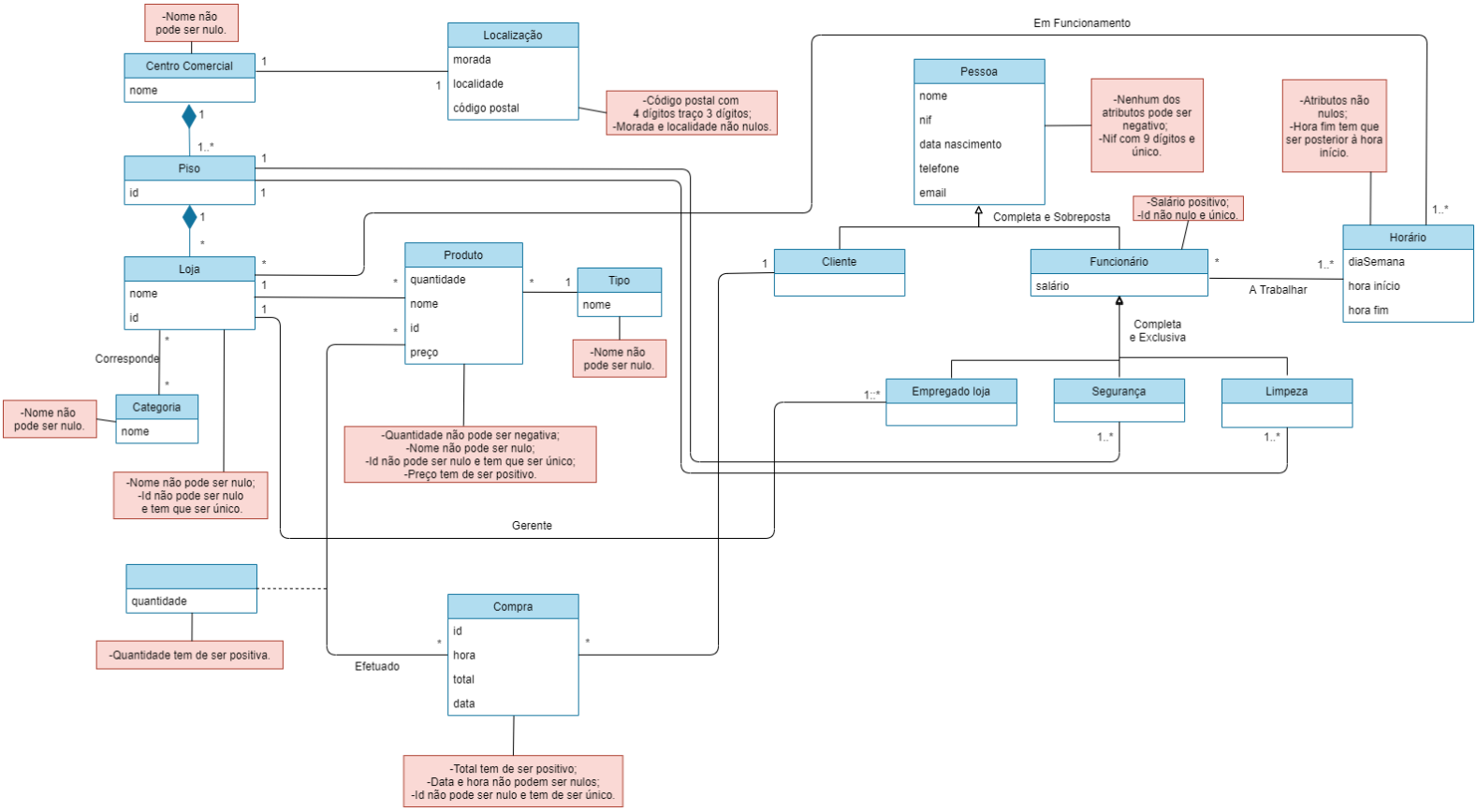
Criámos a classe Horário de forma a representar quando a loja está aberta bem como o horário de cada Funcionário. Do horário guarda-se o dia, hora de início e de fim.

Tal implica um consenso entre o horário de funcionamento de uma Loja e dos seus Empregados de Loja. Desta forma, estando uma loja aberta, tem que haver pelo menos um Empregado de Loja a trabalhar. Esta implicação é representada como uma restrição no nosso gráfico UML.

UML


Localização
morada
localidade
código postal
Centro Comercial
nome
Piso
número
Loja
nome
número
Produto
quantidade
nome
código
preço
Tipo
nome
Categoria
nome
gerente
quantidade
Compra
dia
hora
total
Pessoa
nome
nif
data nascimento
telefone
email
Completa e
Sobreposta
Cliente
Funcionário
salário
Completa e
Exclusiva
Empregado loja
Segurança
Limpeza
Horário
dia
hora início
hora fim
1
1..*
*

UML revisto



Notas: o UML foi novamente alterado para a Entrega 3 mediante as indicações do professor.

Esquema Relacional

- CentroComercial (idCC, nome, idLocal->Localização)
- Localização (idLocal, morada, localidade, códigoPostal)
- Piso (idPiso, idCC->CentroComercial)
- Loja (idLoja, nome, idPiso->Piso, idGerente->Funcionário)
- Categoria (idCategoria, nome)
- Corresponde (idLoja->Loja, idCategoria->Categoria)
- Produto (idProduto, nome, quantidade, preço, idTipo->Tipo, idLoja->Loja)
- Tipo (idTipo, nome)
- Compra (idCompra, data, hora, total, nif->Cliente)
- Efetuado (idEfetuado, idCompra->Compra, idProduto->Produto, quantidade)
- Horário (idHorário, diaSemana, horaInício, horaFim)
- HorárioFuncionário (idHorário->Horário, nif->Funcionário)
- HorárioLoja (idHorário->Horário, idLoja->Loja)
- Pessoa (nif, nome, dataNascimento, telefone, email)
- Cliente (nif->Pessoa)
- Funcionário (nif->Pessoa, salário)
- EmpregadoLoja (nif->Funcionário, idLoja->Loja)
- Segurança (nif->Funcionário, idPiso->Piso)
- Limpeza (nif->Funcionário, idPiso->Piso)

Notas: alterado para a Entrega 3 mediante as indicações do professor.

Análise de dependências funcionais e formas normais

- CentroComercial (idCC, nome, idLocal->Localização)

  FD:

  idCC->nome, idLocal

  nome, idLocal->idCC

  Formas:

  3NF: não viola

  BCNF: não viola

- Localização (idLocal, morada, localidade, códigoPostal)

  FD:

  idLocal->morada, localidade, códigoPostal

  Formas:

  3NF: não viola

  BCNF: não viola

- Piso (idPiso, idCC->CentroComercial)

  FD:

  idPiso-> idCC

  Formas:

  3NF: não viola

  BCNF: não viola

- Loja (idLoja, nome, idPiso->Piso, idGerente->Funcionário)

  FD:

  idLoja->nome, idPiso, idGerente

  nome->idLoja

  Formas:

  3NF: não viola

  BCNF: não viola

- Categoria (idCategoria, nome)

  FD:

  idCategoria->nome

  nome-> idCategoria

  Formas:

  3NF: não viola

  BCNF: não viola

- Corresponde (idLoja->Loja, idCategoria->Categoria)

  FD: --

  Formas:

  3NF: não viola

  BCNF: não viola

- Produto (idProduto, nome, quantidade, preço, idTipo->Tipo, idLoja->Loja)

  FD:

  idProduto->nome, quantidade, preço, idTipo, idLoja

  Formas:

  3NF: não viola

  BCNF: não viola

- Tipo (idTipo, nome)

  FD:

  idTipo->nome

  nome->idTipo

  Formas:

  3NF: não viola

  BCNF: não viola

- Compra (idCompra, data, hora, total, idCliente->Cliente)

  FD:

  idCompra->data, hora, total, idCliente

  data, hora, idCliente, total->idCompra

  Formas:

  3NF: não viola

  BCNF: não viola

- Efetuado (idEfetuado, idCompra->Compra, idProduto->Produto, quantidade)

  FD: idEfetuado->idCompra, idProduto, quantidade

  idCompra, idProduto->idEfetuado, quantidade

  Formas:

  3NF: não viola

  BCNF: não viola

- Horário (idHorário, diaSemana, horaInício, horaFim)

  FD:

  idHorário->diaSemana, horaInício, horaFim

  diaSemana, horaInício, horaFim->idHorário

  Formas:

  3NF: não viola

  BCNF: não viola

- HorárioFuncionário (idHorário->Horário, nif->Funcionário)

  FD: --

  Formas:

  3NF: não viola

  BCNF: não viola

- HorárioLoja (idHorário->Horário, idLoja->Loja)

  FD: --

  Formas:

  3NF: não viola

  BCNF: não viola

- Pessoa (nif, nome, dataNascimento, telefone, email)

  FD:

  nif->nome, dataNascimento, telefone, email

  telefone->nif, nome, dataNascimento, email

  email-> nif, nome, dataNascimento, telefone

  Formas:

  3NF: não viola

  BCNF: não viola

- Cliente (nif->Pessoa)

  FD: --

  Formas:

  3NF: não viola

  BCNF: não viola

- Funcionário (nif->Pessoa, salário)

  FD:

  nif->salário

  Formas:

  3NF: não viola

  BCNF: não viola

- EmpregadoLoja (nif->Funcionário, idLoja->Loja)

  FD:

  nif-> idLoja

  Formas:

  3NF: não viola

  BCNF: não viola

- Segurança (nif->Funcionário, idPiso->Piso)

  FD:

  nif->idPiso

  Formas:

  3NF: não viola

  BCNF: não viola

- Limpeza (nif->Funcionário, idPiso->Piso)

  FD:

  nif->idPiso

  Formas:

  3NF: não viola

  BCNF: não viola

Conclusões:

Uma relação segue BCFC se, para cada não trivial A -> B, A é uma (super)key, ou seja, a partir de A chegamos a todos os atributos.

Uma relação segue 3NF se, para cada não trivial A -> B, A é uma (super)key ou B é um atributo primo, ou seja, faz parte de pelos menos uma chave da relação.

Como a partir da parte esquerda (A) de cada dependência chegamos a todos os atributos (B), conclui-se que A é uma (super)key. Logo, todas as relações da base de dados seguem as 2 formas pedidas: Forma Normal Boyce-Codd e 3ª Forma Normal.

Notas: ligeiramente alterado desde a Entrega 2 mediantes as indicações do professor e devido a algumas alterações no modelo relacional.

Lista de forma de implementação das restrições

- **Centro comercial**

O centro comercial precisa de ter um nome

nome NOT NULL

Não pode haver dois centros comerciais com o mesmo id

idCC PRIMARY KEY

O idLocal deve corresponder a um id de uma Localização

idLocal REFERENCES Localização (idLocal)

- **Localização**

Não podem existir duas localizações com o mesmo id

idLocal PRIMARY KEY

Não pode haver duas localizações com a mesma morada, código postal e localidade

UNIQUE (morada, localidade, códigoPostal)

Todas as localizações têm de ter uma localidade, um código postal e uma morada

morada NOT NULL

localidade NOT NULL

código postal NOT NULL

- **Piso**

Não podem existir pisos com o mesmo id

idPiso PRIMARY KEY

O idCC deve corresponder ao id de um CentroComercial

idCC REFERENCES CC (idCC)

- **Loja**

Não pode haver duas lojas com o mesmo id

idLoja PRIMARY KEY

Todas as lojas têm de ter um nome atribuído e estes têm que ser diferentes, não pode existir 2 lojas com o mesmo nome

nome UNIQUE NOT NULL

O id do Gerente da loja deve corresponder ao nif de um Funcionário

idGerente REFERENCES Funcionário(nif)

O id do Piso deve corresponder ao id de um Piso

idPiso REFERENCES Piso(idPiso)

- **Compra**

Não pode haver duas compras com o mesmo id

idCompra PRIMARY KEY

Data, hora e total não podem ser nulos

dia NOT NULL

hora NOT NULL

total NOT NULL

O total tem de ser maior que zero

total CHECK (total > 0)

O id do Cliente corresponde a um id de um Cliente

idCliente REFERENCES Cliente (idCliente)

- **Categoria**

Não pode haver duas categorias com o mesmo id

idCategoria PRIMARY KEY

O nome da categoria não pode ser nulo e tem de ser único

nome UNIQUE NOT NULL

- **Efetuado**

Não pode haver 2 efetuados com o mesmo id

idEfetuado PRIMARY KEY

Não deve haver duas instâncias com o mesmo par (idCompra, idProduto)

UNIQUE (idCompra, idProduto)

O id da Compra corresponde a um id de Compra

idCompra REFERENCES Loja (idCompra)

O id do Produto corresponde a um id da tabela Produto

idProduto REFERENCES Categoria (idProduto)

A quantidade do Produto comprada tem se ser positiva e não nula

quantidade NOT NULL

quantidade CHECK (quantidade > 0)

- **Corresponde**

Não deve haver duas instâncias com o mesmo par (idLoja, idCategoria)

(idLoja, idCategoria) PRIMARY KEY

O id da Loja corresponde a um id de Loja

idLoja REFERENCES Loja (idLoja)

O id da Categoria corresponde a um id de Categoria

idCategoria REFERENCES Categoria (idCategoria)

- **Tipo**

Não pode haver dois tipos com o mesmo id

idTipo PRIMARY KEY

Nome não pode ser nulo

nome NOT NULL

- **Produto**

Não pode haver dois produtos com o mesmo id

idProduto PRIMARY KEY

A quantidade não pode ser nula nem negativa

quantidade NOT NULL

quantidade CHECK (quantidade>=0)

O preço não pode ser nulo e tem de ser positivo

preco NOT NULL

preco CHECK (preco>0)

Nome não pode ser nulo

nome NOT NULL

O id do Tipo deve corresponder a um id do Tipo

idTipo REFERENCES Tipo (idTipo)

O id da Loja deve corresponder a um id da Loja

idLoja REFERENCES Loja (idLoja)

- **Pessoa**

Não pode haver duas pessoas com o mesmo nif

nif PRIMARY KEY

Todas as pessoas devem ter um nome, data de nascimento, telefone e email

nome NOT NULL

data nascimento NOT NULL

telefone NOT NULL

email NOT NULL

O telefone e o email de uma pessoa têm de ser únicos

telefone UNIQUE

email UNIQUE

- **Cliente**

Não pode haver dois Clientes com o mesmo nif

nif PRIMARY KEY

O nif corresponde a um nif de uma pessoa na tabela Pessoa

nif REFERENCES Pessoa (nif)

- **Funcionário**

Não pode haver dois funcionários com o mesmo nif

nif PRIMARY KEY

O nif corresponde a um nif de uma pessoa na tabela Pessoa

nif REFERENCES Pessoa (nif)

Todos os funcionários têm um salário associado e positivo

salário NOT NULL

salário CHECK (salário > 0)

- **Empregado Loja**

Não pode haver dois Empregados Loja com o mesmo nif

nif PRIMARY KEY

O nif corresponde a um nif de um funcionário da tabela Funcionário

nif REFERENCES Funcionário (nif)

O id da Loja corresponde a um id da tabela Loja

idLoja REFERENCES Loja (idLoja)

- **Segurança**

Não pode haver dois seguranças com o mesmo nif

nif PRIMARY KEY

O nif corresponde a um nif de um funcionário da tabela Funcionário

nif REFERENCES Funcionário (nif)

O id do Piso corresponde a um id da tabela Piso

idPiso REFERENCES Loja (idPiso)

- **Limpeza**

Não pode haver dois funcionários de limpeza com o mesmo nif

nif PRIMARY KEY

O nif corresponde a um nif de um funcionário da tabela Funcionário

nif REFERENCES Funcionário (nif)

O id do Piso corresponde a um id da tabela Piso

idPiso REFERENCES Loja (idPiso)

- **Horário**

Não pode haver dois horários com o mesmo id

idHorário PRIMARY KEY

Deve existir apenas um horário para cada combinação diferente de dia semana, hora início, hora fim

Nenhum dos atributos pode ser nulo

diaSemana NOT NULL

horaInício NOT NULL

horaFim NOT NULL

A hora de fim tem de ser posterior à hora de início

CHECK (hora fim > hora início)

O dia semana tem de ser um dia de semana válido

dia semana CHECK (dia semana = 'SEGUNDA-FEIRA' OR
dia semana = 'TERÇA-FEIRA' OR
dia semana = 'QUARTA-FEIRA' OR
dia semana = 'QUINTA-FEIRA' OR
dia semana = 'SEXTA-FEIRA' OR
dia semana = 'SABADO' OR
dia semana = 'DOMINGO')

- **HorárioFuncionário**

Não pode haver duas instâncias do par idHorário e nif

(idHorário, nif) PRIMARY KEY

O nif do Funcionário deve corresponder a um nif da tabela Funcionário

nif REFERENCES Funcionário (nif)

O id Horário deve corresponder a um id da tabela Horário

idHorário REFERENCES Horário (idHorário)

- **HorárioLoja**

Não pode haver duas instâncias do par idHorário e idLoja

(idHorário, idLoja) PRIMARY KEY

O id da Loja deve corresponder a um id da tabela Loja

idLoja REFERENCES Loja (idLoja)

O id Horário deve corresponder a um id da tabela Horário

idHorário REFERENCES Horário (idHorário)

Lista de Interrogações em Linguagem Natural

1. Lista ordenada de forma decrescente do número de horas completas que cada Funcionário trabalha por semana.
2. Lista de produtos por cada Loja que precisam de ser reabastecidos, ou seja, existe em Loja em quantidade inferior a 10.
3. Média dos salários dos Funcionários de cada Loja.
4. Produtos mais comprados, ordenados de forma decrescente pelo número total de compras.
5. Número de pessoas que fazem compras do CC divididas por idade, menores que 18, entre 18 e 64, e superior a 64 anos.
6. Quais os dias em que o gerente de cada loja trabalha.
7. Qual a loja em cada piso que tem menos funcionários.
8. Quantos dias da semana é que cada segurança com 50 anos ou mais trabalha após as 20 horas.
9. Total de vendas de cada loja, ordenado de forma decrescente.
10. Compra mais cara efetuada por cada cliente.

Lista dos Gatilhos

1. Verifica se os Funcionários têm a idade mínima de 18 anos.
2. Quando é criado um Efetuado é criada uma Compra, caso não exista uma com o id fornecido com o idCompra seguinte ao último inserido, data e hora atuais e total calculado pela quantidade do produto selecionado, o nif é posto a NULL.
3. Atualiza a quantidade de produto existente em Loja após ser feita uma compra.

Observações: Estando limitados pelo número de gatilhos a ser avaliados foram selecionados estes. Em relação ao primeiro gatilho poderiam ser criados outros nomeadamente ao dar *update* da data de nascimento verificar se era uma data válida. Quanto ao segundo gatilho podia também ser criado outro para que sempre que se cria um Efetuado, o total da Compra com o id correspondente seria atualizado. Quanto ao terceiro gatilho poderia ser criado outro que complementasse a restrição da quantidade em Loja de produto ser maior ou igual a zero; não deixando fazer um Efetuado se a quantidade de produto existente fosse zero e também que não fosse criado um Efetuado e uma Compra se a quantidade pretendida fosse superior ao que existe em *stock*.