U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Base de dados sobre

Cursos na área da informática, em Portugal

Unidade Curricular:
Base de dados

Professor:
Michel Ferreira

02LEIC09, grupo 909:

Fábio Almeida Teixeira, up202006345@fe.up.pt

Gustavo Almeida da Costa, up202004187@fe.up.pt

Mafalda Bastos da Costa, up202006417@fe.up.pt

Índice

Contexto

Pretende-se armazenar a informação relativa ao ano letivo 2020/2021, nas Licenciaturas e Mestrados Integrados na área da informática, em Portugal.

De cada **estudante** pretende-se armazenar o seu nome, data de nascimento, sexo e nacionalidade (país de origem). Apesar do curso ser em Portugal, não é obrigatório que os alunos sejam portugueses, podendo vir do estrangeiro, inclusive em regime de Erasmus.

Relativamente aos **cursos** em que cada aluno está inscrito é necessário saber o seu código de identificação, que é único e permite diferenciar entre outros cursos com o mesmo nome, o grau, a propina necessária pagar, a média do último colocado, o número de vagas disponíveis e o número de vagas preenchidas na primeira fase, neste ano letivo. Podendo um aluno estar inscrito em mais do que um curso, para cada curso que frequenta precisamos de saber o ano curricular e a média respetiva.

Alguns trabalhadores estudantes poderão optar por um curso de regime pós-laboral.

Como, para existir um dado curso, é necessário a existência de uma **instituição**, relativamente a esta será necessário armazenar o seu código único, o seu nome, localização, tipo de financiamento, isto é, se é um estabelecimento público ou privado, e tipo de ensino, isto é, se se trata de um Instituto Politécnico ou Universidade.

Uma instituição é composta por várias **divisões** que são responsáveis pela gestão dos vários cursos. Assim sendo, será de relevo armazenar o seu nome, código, que poderá não ser distinto de outras divisões da mesma instituição, uma morada, o número total de cursos que estão sobre a sua alçada e o tipo de divisão (faculdade, departamento, escola superior).

Porque um aluno não recebe formação instantânea, este terá de frequentar várias **unidades curriculares**, das quais o código identificador, nome e período de lecionamento (ano e semestre), deverão ser guardados.

Para um aluno obter aprovação a uma dada unidade curricular este tem de obter uma classificação igual ou superior a 10 valores.

A par disto, os **docentes** de cada unidade curricular e a sua função na divisão em que trabalham, deverão ser armazenados. Um docente poderá pertencer a várias divisões de instituições diversas, ou até da mesma.

Como é permitido a um aluno frequentar vários cursos em simultâneo, assim como um docente lecionar unidades curriculares de vários cursos, é necessário o registo, por parte destes, nas respetivas instituições, através de um **número mecanográfico** único.

Diagrama UML – Entrega I

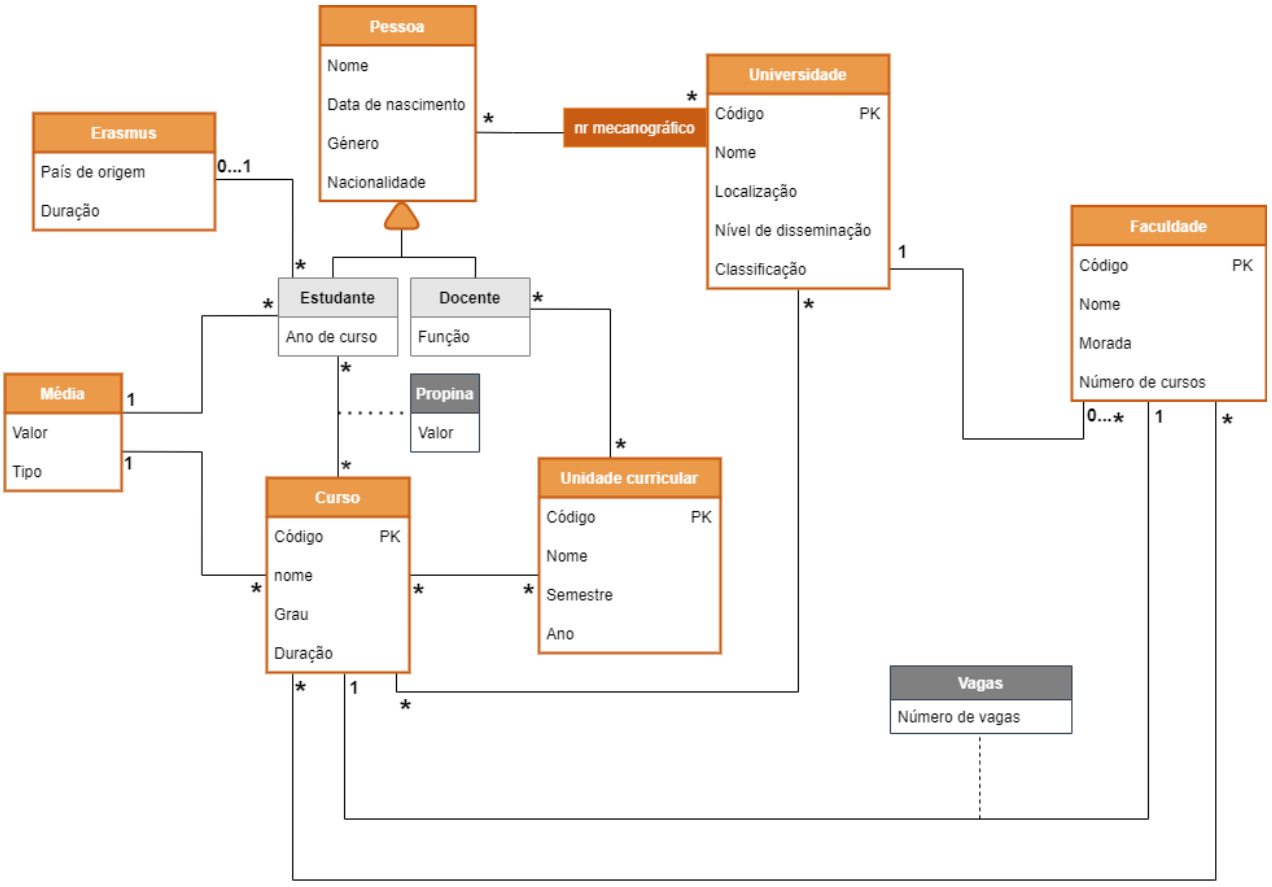
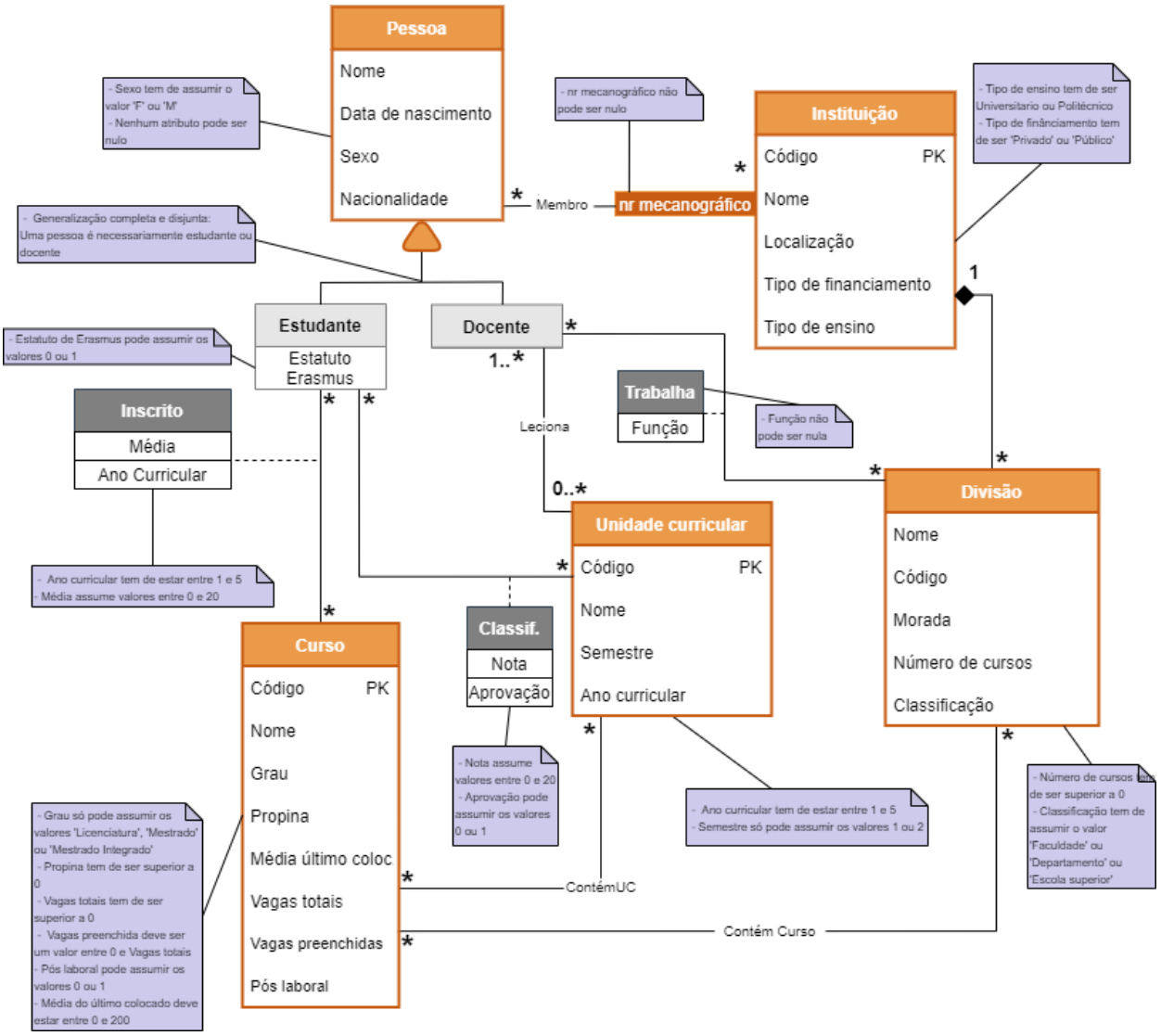


Diagrama UML – Revisto



MUDANÇAS: A fim de aumentar a qualidade do trabalho fizemos várias alterações ao diagrama UML, incluindo aquelas sugeridas pelo professor, como por exemplo:

- Introdução de restrições
- Criação de duas classes de associação (“Inscrito” e “Trabalha”)
- Especificação do tipo de generalização
- Remoção de duas classes (“Média” e “ Erasmus”)
- Criação de várias ligações (“Docente” - “Divisão”, por exemplo)

O contexto também foi alterado para acomodar estas modificações.

Esquema Relacional

Instituicao(codigoInsituicao, nome, localizacao, tipoFinanciamento, tipoEnsino)

Divisao(idDivisao, codigoDivisao, nome, morada, numCursos, classificacao, codigoInstituicao → Instituicao)

UnidadeCurricular(codigoUC, nome, semestre, anoCurricular)

Curso(codigoCurso, nome, grau, propina, mediaUltimoColocado, vagasTotais, vagasPreenchidas, posLaboral)

Pessoa(idPessoa, nome, dataNascimento, sexo, nacionalidade)

Estudante(idPessoa → Pessoa, estatutoErasmus)

Docente(idPessoa → Pessoa)

Membro(codigoInstituicao → Instituicao, idPessoa → Pessoa, nrMecanografico)

Leciona(idPessoa → Docente, codigoUC → UnidadeCurricular)

Classificacao(idPessoa → Estudante, codigoUC → UnidadeCurricular, nota, aprovacao)

InscritoCurso(idPessoa → Estudante, codigoCurso → Curso, media, anoCurricular)

ContemUC(codigoCurso → Curso, codigoUC → UnidadeCurricular)

ContemCurso(idDivisao → Divisao, codigoCurso → Curso)

Trabalha(idPessoa → Docente, idDivisao → Divisao, funcao)

Análise de Dependências Funcionais e Formas Normais

Dependências Funcionais (FDs):

Instituicao*(codigoInsituicao, nome, localizacao, tipoFinanciamento, tipoEnsino)*

- códigoInstituicao → nome, localizacao, tipoFinanciamento, tipoEnsino
- nome → codigoInstituicao, localizacao, tipoFinanciamento, tipoEnsino

Divisao*(idDivisao, codigoDivisao, nome, morada, numCursos, classificacao, codigoInstituicao → Instituicao)*

- idDivisao → codigoDivisao, nome, morada, numCursos, classificacao, codigoInstituicao
- codigoDivisao, nome → idDivisao, numCursos, classificacao, codigoInstituicao

UnidadeCurricular*(codigoUC,nome, semestre, anoCurricular)*

- codigoUC → nome, semestre, anoCurricular

Curso*(codigoCurso, nome, grau, propina, mediaUltimoColocado, vagasTotais, vagasPreenchidas, posLaboral)*

- codigoCurso → nome, grau, propina, mediaUltimoColocado, vagasTotais, vagasPreenchidas, posLaboral

Pessoa*(idPessoa, nome, dataNascimento, sexo, nacionalidade)*

- idPessoa → nome, dataNascimento, sexo, nacionalidade

Estudante*(idPessoa, estatutoErasmus)*

- idPessoa → estatutoErasmus

Docente*(idPessoa)*

- não tem FDs.

Membro*(codigoInstituicao → Instituicao, idPessoa → Pessoa, nrMecanografico)*

- codigoInstituicao, idPessoa → nrMecanografico

Leciona*(idPessoa →Docente, codigoUC →UnidadeCurricular)*

- Não tem FDs.

Classificacao*(idPessoa →Estudante, codigoUC →UnidadeCurricular, nota, aprovacao)*

- idPessoa, codigoUC → nota, aprovacao
- nota → aprovacao

InscritoCurso*(idPessoa →Estudante, codigoCurso →Curso)*

- Não tem FDs.

ContemUC*(codigoCurso →Curso, codigoUC →UnidadeCurricular)*

- Não tem FDs.

ContemCurso*(idDivisao → Divisao, codigoCurso →Curso)*

- Não tem FDs.

Trabalha*(idPessoa →Docente, idDivisao → Divisao, funcao)*

- Não tem FDs.

Forma Normal de Boyce-Codd (BCNF):

Uma relação R encontra-se na BCNF se, para cada FD A → B:
- A → B é trivial (B é um subconjunto de atributos de A)
- A é uma (super)chave (A determina todos os atributos de R)

3ª Forma Normal (3NF):

Uma relação R encontra-se na 3NF se, para cada FD A → B não trivial:
- A é uma (super)chave
- B consiste apenas em atributos primos (sendo atributos primos aqueles que pertencem a qualquer chave da relação R)

Assim sendo, a relação "Classificacao(idPessoa → Estudante, codigoUC → UnidadeCurricular, nota, aprovacao)" não segue a BCNF nem a 3NF, pelo que é necessário dividir em duas relações:

Aprovacao(nota, aprovado)

- nota → aprovado

Classificacao(idPessoa → Estudante, codigoUC → UnidadeCurricular, nota)

- idPessoa, codigoUC → nota

Sobre as restantes relações da base de dados, podemos concluir que respeitam a Forma Normal de Boyce-Codd (BCNF), tal como a 3ª forma normal (3NF).

Restrições

Instituicao

- Não podem haver duas instituições com o mesmo 'codigoInstituicao':
  - codigoInstituicao PRIMARY KEY
- O atributo 'nome' é único
- Instituicao só pode ter tipo de financiamento 'Publico' ou 'Privado':
  - CONSTRAINT ValidTipoFinanciamento CHECK(tipoFinanciamento = 'Privado' OR tipoFinanciamento = 'Publico')
- Instituicao só pode ser do tipo 'Universitario' ou 'Politecnico':
  - CONSTRAINT ValidTipoEnsino CHECK(tipoEnsino = 'Universitario' OR tipoEnsino = 'Politecnico')
- Nenhum atributo pode ser nulo.

Divisao

- Não podem haver duas divisões com o mesmo 'idDivisao':
  - idDivisao PRIMARY KEY
- O 'numCursos' tem de ser superior a 0 e tem como default o valor 0:
  - CONSTRAINT ValidNumCursos CHECK(numCursos > 0)
  - numCursos DEFAULT 0
- A 'classificacao' deve assumir o valor 'Faculdade' ou 'Departamento':
  - CONSTRAINT ValidClassificacao CHECK(classificacao = 'Faculdade' OR classificacao = 'Departamento' OR classificacao = 'Escola Superior')
- Nenhum atributo pode ser nulo.

UnidadeCurricular

- Não podem haver duas unidades curriculares com o mesmo atributo 'codigoUC':
  - codigoUC PRIMARY KEY
- 'anoCurricular' só pode ter valores entre 1 e 5:
  - CONSTRAINT ValidAnoCurricular CHECK(anoCurricular BETWEEN 0 AND 5)
- 'semestre' só pode ter valores 1 ou 2:
  - CONSTRAINT ValidSemestre CHECK(semestre = 1 or semestre = 2)
- Nenhum atributo pode ser nulo.

Curso

- Não podem haver dois cursos com o mesmo 'codigoCurso':
  - codigoCurso PRIMARY KEY
- 'grau' deve assumir o valor 'Licenciatura', 'Mestrado Integrado' e tem como default o valor 'Licenciatura':
  - CONSTRAINT ValidGrau CHECK(grau in ('Licenciatura', 'Mestrado Integrado'))
  - grau DEFAULT 'Licenciatura'

- O valor de ‘propina’ deve ser superior a 0:
    - CONSTRAINT ValidPropina CHECK(propina >= 0)
- ‘mediaUltimoColocado’ deve assumir valores entre 0 e 200:
    - CONSTRAINT ValidmediaUltCol CHECK(mediaUltimoColocado BETWEEN 0 and 200)
- O valor de ‘vagasTotais’ deve ser superior a 0 ou NULL:
    - CONSTRAINT ValidVagasTotais CHECK(vagasTotais > 0 or vagasTotais is NULL)
- O valor de ‘vagasPreenchidas’ deve estar entre 0 e ‘vagasTotais’ ou ser NULL:
    - CONSTRAINT ValidVagasPreenchidas CHECK(vagasPreenchidas IS NULL or vagasPreenchidas BETWEEN 0 AND vagasTotais)
- ‘posLaboral’ só pode assumir os valores 0 ou 1, sendo 0 por default:
    - CONSTRAINT ValidPosLaboral CHECK(posLaboral = 0 or posLaboral = 1)
    - posLaboral DEFAULT 0
- Os seguintes atributos não podem ser nulos:
    - nome, posLaboral

Pessoa

- Duas pessoas não podem ter o mesmo ‘idPessoa’:
    - idPessoa PRIMARY KEY
- O atributo ‘sexo’ só pode assumir os valores ‘M’ ou ‘F’:
    - CONSTRAINT ValidSexo CHECK(sexo = 'M' OR sexo = 'F')
- O atributo ‘nacionalidade’ assume o valor ‘PT’ por default:
    - DEFAULT 'PT'
- Nenhum atributo pode ser nulo.

Estudante

- Dois estudantes não podem ter o mesmo ‘idPessoa’:
    - idPessoa PRIMARY KEY
- O atributo ‘estatutoErasmus’ só pode assumir os valores 0 ou 1, sendo 0 por deafult:
    - CONSTRAINT ValidEstatutoErasmus CHECK(estatutoErasmus = 0 OR estatutoErasmus = 1)
    - estatutoErasmus DEFAULT 0

Docente

- Dois docentes não podem ter o mesmo ‘idPessoa’:
    - idPessoa PRIMARY KEY

Membro

- Não podem haver dois membros com o mesmo par ‘codigoInstituicao’, ‘idPessoa’:
    - PRIMARY KEY(‘codigoInstituicao’, ‘idPessoa’)
- Nenhum atributo pode ser nulo.

Leciona

- Não podem haver duas instâncias com o mesmo par 'idPessoa' , 'codigoUC':
    - PRIMARY KEY('idPessoa' , 'codigoUC')

Classificacao

- Não podem haver duas classificações com o mesmo par 'idPessoa' , 'codigoUC':
    - PRIMARY KEY('idPessoa' , 'codigoUC')
- 'nota' deve assumir valores entre 0 e 20:
    - CONSTRAINT ValidNota CHECK(nota BETWEEN 0 AND 20)
- Nenhum atributo pode ser nulo.

Aprovacao

- Não podem haver duas aprovações diferentes com a mesma 'nota':
    - nota PRIMARY KEY
- 'aprovado' só pode assumir o valor 0 ou 1, sendo 1 sempre que nota for superior a 10:
    - CONSTRAINT ValidAprov CHECK(nota => 10 and aprovacao = 1 or nota < 10 and aprovacao = 0)
- Nenhum atributo pode ser nulo.

InscritoCurso

- Não podem haver duas instâncias com o mesmo par ('idPessoa', 'codigoCurso'):
    - PRIMARY KEY('idPessoa' , 'codigoCurso')
- 'anoCurricular' só pode ter valores entre 1 e 5:
    - CONSTRAINT ValidAnoCurricular CHECK(anoCurricular BETWEEN 1 AND 5)
- 'media' só pode ter valores entre 0 e 20:
    - CONSTRAINT ValidMedia CHECK(media BETWEEN 0 AND 20)
- 'anoCurricular' não pode ser nulo.

ContemUc

- Não podem haver duas instâncias com o mesmo par ('codigoCurso', 'codigoUC'):
    - PRIMARY KEY('codigoCurso', 'codigoUC')

ContemCurso

- Não podem haver duas instâncias com o mesmo par ('codigoCurso', 'idDivisao'):
    - PRIMARY KEY('codigoCurso', 'idDivisao')

Trabalha

- Não podem haver duas instâncias com o mesmo par ('idPessoa', 'idDivisao'):
    - PRIMARY KEY('idPessoa', 'idDivisao')
- Nenhum atributo pode ser nulo.

Interrogações

1. Informações dos alunos com estatuto de Erasmus.

2. Curso com maior média do último colocado.

3. Diretores de curso de instituições universitárias.

4. Cursos de carácter pós-laboral em instituições públicas.

5. Média das Unidades Curriculares relacionadas com ‘Algebra’.

6. Docentes que lecionam Unidades Curriculares com ‘Bases de Dados’ no nome, na Universidade do Porto.

7. Aluno com maior média em cada curso, por ano curricular.

8. Cursos com vagas por preencher e o número de vagas respetivo.

9. Alunos, e respetivo número mecanográfico, que reprovaram a mais de 2 Unidades Curriculares e a quantas reprovaram.

10. Número e taxa de alunos do sexo feminino e masculino, por curso e total.

Gatilhos

1. Quando é adicionado ou removido um curso a uma divisão, o número de cursos da mesma é, respetivamente, incrementado ou decrementado.

2. Impede a inserção de docentes cuja idade seja inferior a 21 anos, assim como a sua atualização para valores “ilegais”.

3. Alteração da média de um aluno quando é adicionada, alterada ou removida uma classificação referente a uma Unidade Curricular, num determinado curso.

NOTA 1: Os gatilhos devem ser inseridos na base de dados antes do povoamento da mesma, de modo a garantir o funcionamento das interrogações. Foi incluído um ficheiro “run.sql” que faz a leitura de todos os ficheiros simultaneamente (criar.sql, gatilhoN_adiciona.sql, povoar.sql).

NOTA 2: Foram feitas pequenas alterações nos documentos *criar.sql* e *povoar.sql*, de modo a poder mostrar, de maneira mais clara, as interrogações e gatilhos criados.

Modificações e respetivas razões:

- Colocado um *default value* = 0, no atributo “media” da tabela “InscritoCurso”. O gatilho número 3, deste modo, é ativado e calcula a média automaticamente.
- Colocado *default value* = 0 também no atributo “numCursos” da tabela “Divisao”, para o gatilho 1.
- Mudança, no *povoar.sql*, dos atributos “funcao” da tabela “Trabalha”, anteriormente iguais a ‘Diretor do Curso’, para ‘Diretor de Curso’. Facilita a implementação da interrogação 10.