UPskill

Digital Skills & Jobs

SQL - DDL

BASE DE DADOS – JAVA.NET

P.PORTO

apdc
ASSOCIAÇÃO PORTUGUESA PARA O DESENVOLVIMENTO DAS COMUNICAÇÕES

INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL

CONSELHO COORDENADOR DOS INSTITUTOS SUPERIORES POLITÉCNICOS

Contéudos

- **Modelo Físico**
  - Implementação
- **Criação Tabelas**
  - Restrições Integridade (RI)
- **Alteração Tabelas**
- **Eliminação Tabelas**
- **Exemplos**

UPskill

Digital Skills & Jobs

Modelo Físico

Modelo Físico

- Extraído do Modelo Lógico
- As relações são promovidas a tabelas
- Cada atributo da relação é representado numa coluna da tabela
- Cada coluna da tabela fica associado um domínio existente no Sistema Gestor de Base de Dados (SGBD)
- As restrições de integridade são implementadas de acordo com os mecanismo existentes no SGBD
- É completamente dependente do SGBD selecionado!

Modelo Físico - Implementação

- **Via comandos SQL**
- **SQL**
  - *Structured Query Language*
  - Usada em bases de dados relacionais
  - Divisível em:
    - *Data Definition Language* (DDL) – definição da estrutura e do controlo de acesso aos dados, *e.g.* CREATE TABLE, ALTER TABLE
    - *Data Manipulation Language* (DML) – acesso e manipulação de dados – INSERT, UPDATE, DELETE, SELECT
    - *Data Control Language* (DCL) – definição de permissões de acesso a objetos da base de dados – *e.g.* GRANT, REVOKE
    - *Transaction Control Language* (TCL) – manipulação de transações – *e.g.* COMMIT, ROLLBACK

Modelo Físico - Implementação

- **Criação de tabelas**
  - CREATE TABLE
  - Definição da estrutura da tabela
  - Especificação das restrições associadas à tabela
- **Alteração de tabelas**
  - ALTER TABLE
  - Alteração da estrutura da tabela – adição ou eliminação de campos
  - Alteração das restrições da tabela – adição ou eliminação de RI
- **Eliminação de tabelas**
  - DROP TABLE

UPskill

Digital Skills & Jobs

Criação Tabelas

Modelo Físico – Criação tabelas

```
CREATE TABLE nome_tabela(
        {nome_coluna domínio_coluna
            [{CONSTRAINT definições_RI_coluna}] },
        [{CONSTRAINT definições_RI_tabela}]
)
```

- **Onde domínio_coluna pode ser:**
  - NUMBER, INTEGER, FLOAT
  - CHAR(dim)
  - VARCHAR2(dim)
  - DATE
  - *etc.*

Modelo Físico – Criação tabelas

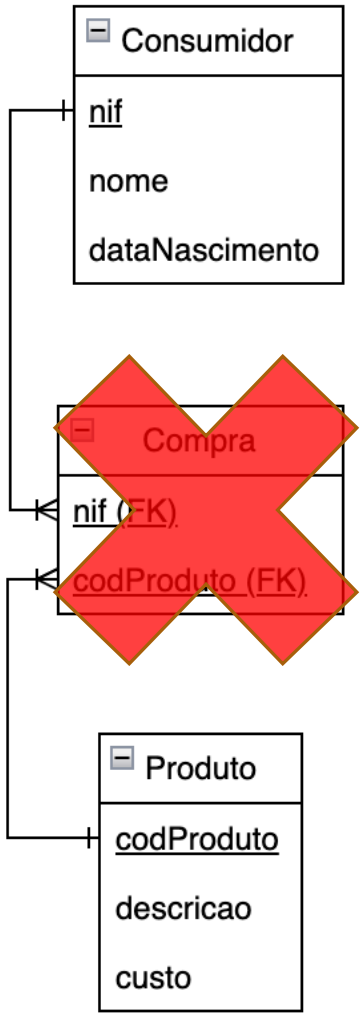
```
CREATE TABLE nome_tabela(
        {nome_coluna domínio_coluna
            [{CONSTRAINT definições_RI_coluna}] },
        [{CONSTRAINT definições_RI_tabela}]
)
```

- **Onde definições_RI podem ser aplicadas a:**
  - Coluna (*inline*) – definição de regras associadas ao valor da coluna do tuplo
  - Tabela (*out-of-line*) – definição de regras associadas a valores de um conjunto de colunas de um tuplo da tabela, *e.g.* definição da PK ou verificação entre duas colunas da tabela

Modelo Físico – Restrições Integridade

- **Podem ser definidas as seguintes RI:**
  - PRIMARY KEY (PK)
  - FOREIGN KEY (FK)
  - NOT NULL (NN)
  - UNIQUE (UK)
  - CHECK (CK)
  - DEFAULT – apenas para coluna
- **Boas práticas:**
  - A PK deve ser definida como RI de tabela
  - A FK deve ser definida numa instrução ALTER TABLE
  - As demais RI devem ser definidas de acordo com a respetiva aplicabilidade, *i.e.* coluna ou tabela
  - Deve cada RI ter uma designação associada, para facilitar a gestão da mesma. No entanto, a RI *default* não permite associar uma designação

Modelo Físico – Criar Tabela (Exemplo)




```
CREATE TABLE consumidor (
        nif          NUMBER,
        nome         VARCHAR2(40)
                CONSTRAINT nn_consumidor_nome NOT NULL,
        datanascimento DATE
                CONSTRAINT nn_consumidor_datanascimento NOT NULL,
        CONSTRAINT pk_consumidor_nif PRIMARY KEY ( nif )
)
```

```
CREATE TABLE produto (
        codproduto NUMBER,
        descricao  VARCHAR(40) DEFAULT 'sem desc'
                CONSTRAINT nn_produto_descricao NOT NULL,
        custo      FLOAT
                CONSTRAINT nn_produto_custo NOT NULL
                CONSTRAINT ck_produto_custo CHECK ( custo > 0 ),
        CONSTRAINT pk_produto_codproduto PRIMARY KEY ( codproduto )
)
```

UPskill

Digital Skills & Jobs

Alteração Tabelas

Modelo Físico – Alteração tabelas (Coluna)

```
ALTER TABLE nome_tabela(
        [ADD nome_coluna domínio_coluna
                [{CONSTRAINT definições_RI_coluna}]]
        [DROP COLUMN nome_coluna]
        [MODIFY nome_coluna [domínio_coluna]
                [{CONSTRAINT definições_RI_coluna}]]
)
```

- **Este comando permite a:**
  - Adição de uma nova coluna com as RI especificadas
  - Eliminação de uma coluna existente
  - Alteração de algumas características da coluna

Modelo Físico –Alterar Tabela (Exemplo)

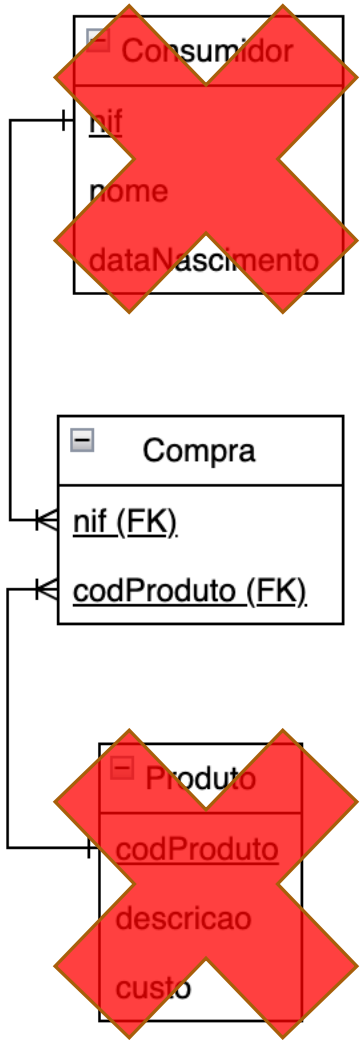
- **Qual é o resultado da execução de cada um dos comandos?**

```
ALTER TABLE consumidor ADD nrcc NUMBER
        CONSTRAINT uk_consumidor_nrcc UNIQUE
        CONSTRAINT nn_consumidor_nrcc NOT NULL

ALTER TABLE consumidor MODIFY
        nrcc
                CONSTRAINT ck_consumidor_nrcc CHECK ( nrcc > 0 )

ALTER TABLE consumidor DROP COLUMN nrcc CASCADE CONSTRAINTS
```

Modelo Físico – Alterar Tabela (Exemplo)




```
CREATE TABLE compra (
        nif       NUMBER,
        codproduto NUMBER,
        CONSTRAINT pk_compra_nif_codprodutor PRIMARY KEY ( nif,
                                                codproduto )
)

ALTER TABLE compra
        ADD CONSTRAINT fk_compra_nif FOREIGN KEY ( nif )
                REFERENCES consumidor ( nif )

ALTER TABLE compra
        ADD CONSTRAINT fk_compra_codproduto FOREIGN KEY ( codproduto )
                REFERENCES produto ( codproduto )
```

UPskill

Digital Skills & Jobs

Eliminação Tabelas

Modelo Físico – Eliminar Tabelas

- **DROP TABLE nome_tabela [CASCADE CONSTRAINTS] [PURGE]**

- **Com este(s) comando(s) pode-se eliminar uma tabela. No entanto:**
- **A opção CASCADE CONSTRAINTS é necessária para eliminar as RI associadas à PK da tabela**
- **A opção PURGE liberta o espaço ocupado pela tabela**

Perguntas

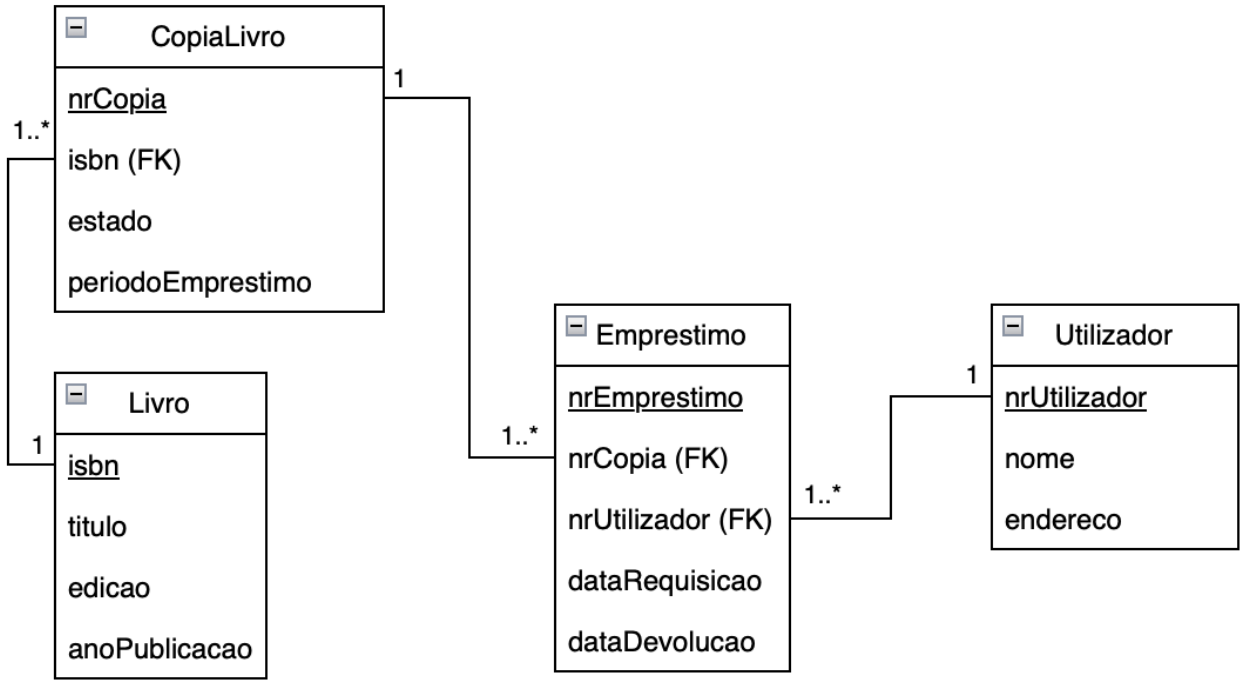
UPskill

Digital Skills & Jobs

Exemplos

Exercício 1

- **Implemente o Modelo Físico do Modelo Relacional do Exercício 1, incluindo as RI.**



Exercício 1

- **Restrições de Integridade**

Relação	Atributo	Restrição
Livro	anoPublicaco	Inferior ou igual ao ano atual
CopiaLivro	estado	Deve ser um: 0 - emprestado 1 - disponível
Emprestimo	dataRequisicao	Inferior ou igual à data atual
	dataDevolucao	Superior ao valor da dataRequisicao
	Uma cópia não pode estar emprestada em datas que já se encontre emprestada	

Exercício 1 – Resolução (1/3)

```
CREATE TABLE livro (
        isbn   NUMBER,
        titulo VARCHAR2(40)
                CONSTRAINT nn_livro_titulo NOT NULL,
        edicao NUMBER
                CONSTRAINT ck_livro_edicao CHECK ( edicao > 1 ),
        ano    DATE --verificar a validade do ano terá que ser com um trigger
                CONSTRAINT nn_livro_ano NOT NULL,

        CONSTRAINT pk_livro_isbn PRIMARY KEY ( isbn )
);

CREATE TABLE copialivro (
        nrcopia         NUMBER
                GENERATED BY DEFAULT AS IDENTITY ( START WITH 1 INCREMENT BY 1 ),
        isbn            NUMBER,
        estado          NUMBER DEFAULT 1
                CONSTRAINT nn_copialivro_estado NOT NULL
                CONSTRAINT ck_copialivro_estado CHECK ( estado IN ( 0, 1 ) ),
        periodoemprestimo NUMBER DEFAULT 7
                CONSTRAINT nn_copialivro_periodoemprestimo NOT NULL
                CONSTRAINT ck_copialivro_periodoempresttimo CHECK ( periodoemprestimo > 0 ),

        CONSTRAINT pk_copialivro_nrcopia PRIMARY KEY ( nrcopia )
);
```

Exercício 1 – Resolução (2/3)

```
CREATE TABLE emprestimo (
        nremprestimo   NUMBER
                GENERATED BY DEFAULT AS IDENTITY ( START WITH 10000 INCREMENT BY 1 ),
        nrcopia       NUMBER
                CONSTRAINT nn_emprestimo_nrcopia NOT NULL,
        nrutilizador   NUMBER
                CONSTRAINT nn_emprestimo_nrutilizador NOT NULL,
        datarequisicao DATE
        --verificar data requisição terá que ser com um trigger
                CONSTRAINT nn_emprestimo_datarequisicao NOT NULL,
        datadevolucao  DATE -- pode ser NULL e/ou poderá haver um trigger que valide a data de devolução,

        CONSTRAINT pk_emprestimo_nremprestimo PRIMARY KEY ( nremprestimo )
);

CREATE TABLE utilizador (
        nrutilizador NUMBER
                GENERATED BY DEFAULT AS IDENTITY ( START WITH 100 INCREMENT BY 1 ),
        nome        VARCHAR(50)
                CONSTRAINT nn_utilizador_nome NOT NULL,
        morada      VARCHAR(100),

        CONSTRAINT pk_utilizador_nrutilizador PRIMARY KEY ( nrutilizador )
);
```

Exercício 1 – Resolução (3/3)

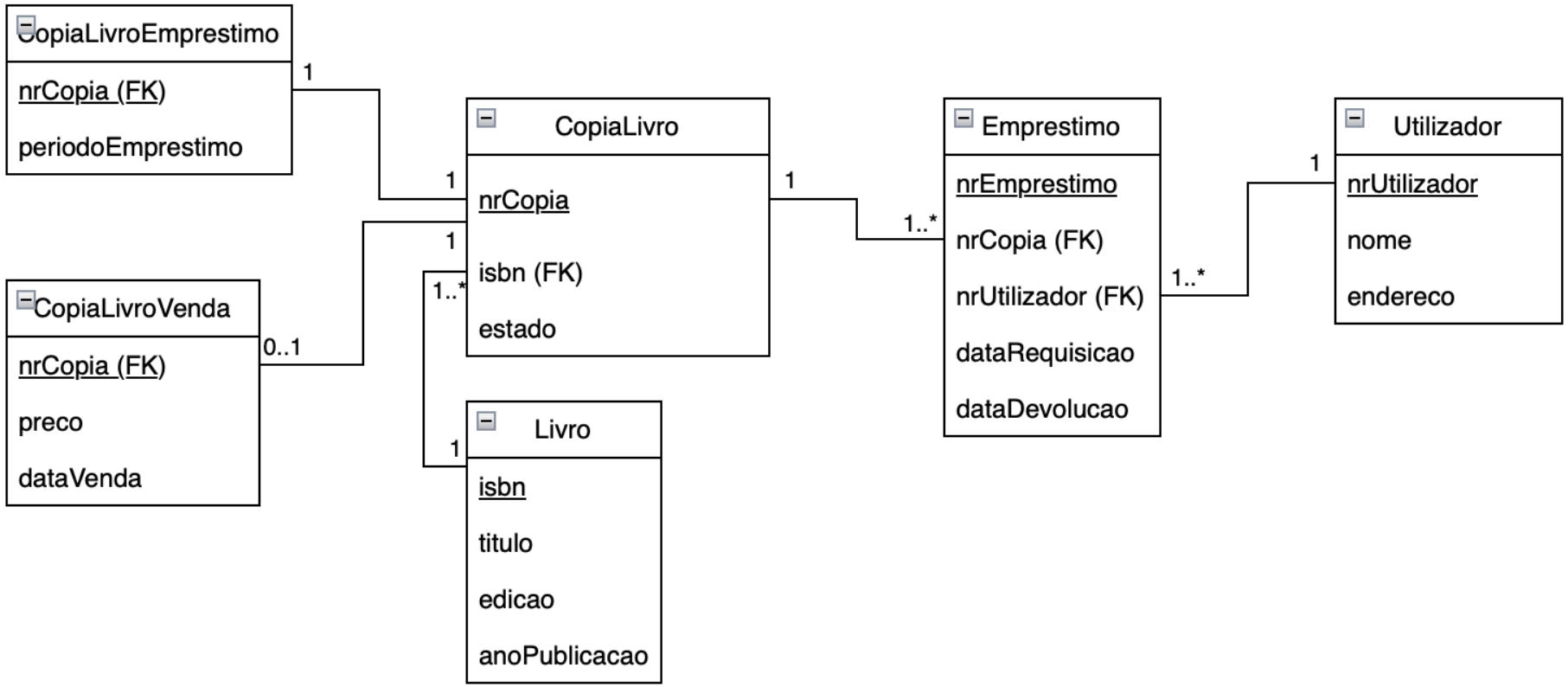
```
ALTER TABLE copialivro
        ADD CONSTRAINT fk_copialivro_isbn FOREIGN KEY ( isbn )
                REFERENCES livro ( isbn );

ALTER TABLE emprestimo
        ADD CONSTRAINT fk_emprestimo_nrcopia FOREIGN KEY ( nrcopia )
                REFERENCES copialivro ( nrcopia );

ALTER TABLE emprestimo
        ADD CONSTRAINT fk_emprestimo_nrutilizador FOREIGN KEY ( nrutilizador )
                REFERENCES utilizador ( nrutilizador );
```

Exercício 2

- **Implemente o Modelo Físico do Modelo Relacional do Exercício 2, incluindo as RI.**



Exercício 2

- **Restrições de Integridade**

Relação	Atributo	Restrição
Livro	anoPublicaco	Inferior ou igual ao ano atual
CopiaLivro	estado	Deve ser um: 0 – emprestado 1 – disponível 2 – extraviado 3 – venda
CopiaLivroVenda	dataVenda	Garantir que não coincide com uma data de empréstimo
Emprestimo	dataRequisicao	Inferior ou igual à data atual
	dataDevolucao	Superior ao valor da dataRequisicao
	Só cópias de livros que não constam da relação CopiaLivroVenda é que podem ser emprestadas. Exceto se dataDevolucao for inferior a dataVenda	

Exercício 2 – Resolução (1/4)

```
CREATE TABLE livro (
        isbn    NUMBER,
        titulo VARCHAR2(40)
                CONSTRAINT nn_livro_titulo NOT NULL,
        edicao NUMBER
                CONSTRAINT ck_livro_edicao CHECK ( edicao > 1 ),
        ano     DATE
        --verificar a validade do ano terá que ser com um trigger
                CONSTRAINT nn_livro_ano NOT NULL,

        CONSTRAINT pk_livro_isbn PRIMARY KEY ( isbn )
);

CREATE TABLE copialivro (
        nrcopia NUMBER
                GENERATED BY DEFAULT AS IDENTITY ( START WITH 1 INCREMENT BY 1 ),
        isbn    NUMBER,
        estado  NUMBER DEFAULT 1
                CONSTRAINT nn_copialivro_estado NOT NULL
                CONSTRAINT ck_copialivro_estado CHECK ( estado BETWEEN 0 AND 3 ),

        CONSTRAINT pk_copialivro_nrcopia PRIMARY KEY ( nrcopia )
);
```

Exercício 2 – Resolução (2/4)

```
CREATE TABLE copialivroemprestimo (
        nrcopia         NUMBER,
        periodoemprestimo NUMBER DEFAULT 7
                CONSTRAINT nn_copialivroemprestimo_periodoemprestimo NOT NULL
                CONSTRAINT ck_copialivroemprestimo_periodoempresttimo CHECK ( periodoemprestimo > 0 ),

        CONSTRAINT pk_copialivroemprestimo_nrcopia PRIMARY KEY ( nrcopia )
);

CREATE TABLE copialivrovenda (
        nrcopia   NUMBER,
        preco     NUMBER
                CONSTRAINT nn_copialivrovenda_preco NOT NULL
                CONSTRAINT ck_copialivrovenda_preco CHECK ( preco > 0 ),
        datavenda DATE
        --verificar data venda terá que ser com um trigger
                CONSTRAINT nn_copialivrovenda_datavenda NOT NULL,

        CONSTRAINT pk_copialivrovenda_nrcopia PRIMARY KEY ( nrcopia )
);
```

Exercício 2 – Resolução (3/4)

```
CREATE TABLE emprestimo (
        nremprestimo   NUMBER
                GENERATED BY DEFAULT AS IDENTITY ( START WITH 10000 INCREMENT BY 1 ),
        nrcopia       NUMBER
        -- garantir que a cópia é de empréstimo terá que ser feito com um trigger
                CONSTRAINT nn_emprestimo_nrcopia NOT NULL,
        nrutilizador   NUMBER
                CONSTRAINT nn_emprestimo_nrutilizador NOT NULL,
        datarequisicao DATE
        --verificar data requisição terá que ser com um trigger
                CONSTRAINT nn_emprestimo_datarequisicao NOT NULL,
        datadevolucao  DATE
        -- pode ser NULL e/ou poderá haver um trigger que valide a data de devolução
        ,

        CONSTRAINT pk_emprestimo_nremprestimo PRIMARY KEY ( nremprestimo )
);
```

Exercício 2 – Resolução (4/4)

```
CREATE TABLE utilizador (
        nrutilizador NUMBER
                GENERATED BY DEFAULT AS IDENTITY ( START WITH 100 INCREMENT BY 1 ),
        nome        VARCHAR(50)
                CONSTRAINT nn_utilizador_nome NOT NULL,
        morada      VARCHAR(100),

        CONSTRAINT pk_utilizador_nrutilizador PRIMARY KEY ( nrutilizador )
);
ALTER TABLE copialivro
        ADD CONSTRAINT fk_copialivro_isbn FOREIGN KEY ( isbn )
                REFERENCES livro ( isbn );
ALTER TABLE copialivroemprestimo
        ADD CONSTRAINT fk_copialivroemprestimo_nrcopia FOREIGN KEY ( nrcopia )
                REFERENCES copialivro ( nrcopia );
ALTER TABLE copialivrovenda
        ADD CONSTRAINT fk_copialivrovenda_nrcopia FOREIGN KEY ( nrcopia )
                REFERENCES copialivro ( nrcopia );
ALTER TABLE emprestimo
        ADD CONSTRAINT fk_emprestimo_nrcopia FOREIGN KEY ( nrcopia )
                REFERENCES copialivro ( nrcopia );
ALTER TABLE emprestimo
        ADD CONSTRAINT fk_emprestimo_nrutilizador FOREIGN KEY ( nrutilizador )
                REFERENCES utilizador ( nrutilizador );
```