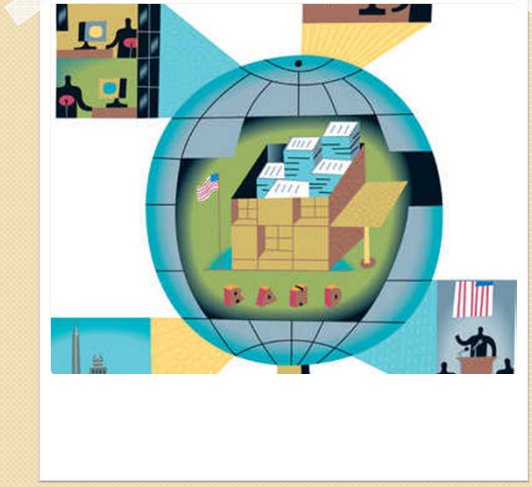
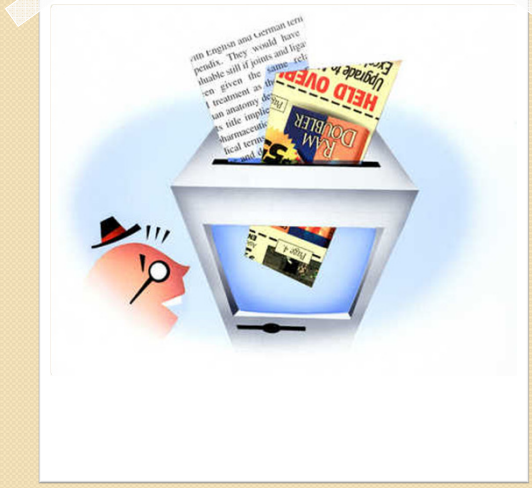


Banco de Dados



Linguagem SQL

A linguagem SQL: história

- Em junho de 1970, o matemático Edgar Frank Codd, publicou o artigo "A Relational Model of Data for Large Shared Data Banks" na revista "Communications of the ACM";
- Neste trabalho, Codd estabeleceu princípios sobre gerencia de banco de dados, denominando-os com o termo relacional;
- Esse material é um marco na área de banco de dados;
- Codd faleceu em 18 de abril de 2003, aos 79 anos;

- A razão do sucesso dos bancos de dados relacionais e da linguagem SQL se deve ao fato de existir um modelo matemático formal que serviu de base para seu desenvolvimento.

A linguagem SQL: história

- A linguagem SQL foi desenvolvido no início dos anos 70 nos laboratórios da IBM em San Jose, dentro do projeto System R, que tinha por objetivo demonstrar a viabilidade da implementação do modelo relacional proposto por E. F. Codd.
- O nome original da linguagem era SEQUEL, acrônimo para "Structured *English* Query Language", vindo daí o fato de, até hoje, a sigla, em inglês, ser comumente pronunciada "síquel".
- A linguagem SQL é um grande padrão de banco de dados. Isto decorre da sua simplicidade e facilidade de uso;
- A SQL é uma linguagem declarativa, em oposição a outras linguagens procedurais.
  - A linguagem SQL especifica a forma do resultado e não o caminho para chegar a ele. Isto reduz o ciclo de aprendizado daqueles que se iniciam na linguagem.

A linguagem SQL: história

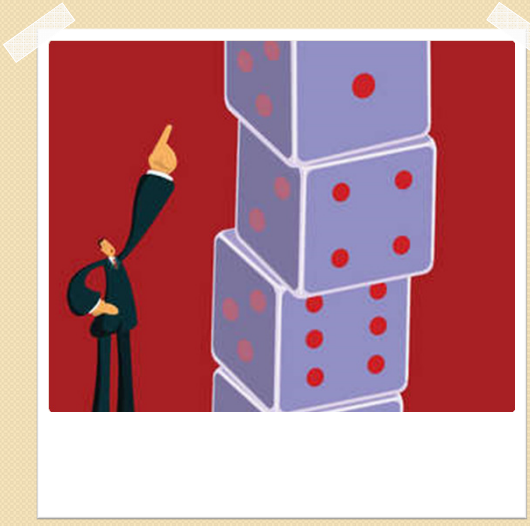
- Embora o SQL tenha sido originalmente criado pela IBM, rapidamente surgiram vários "dialetos" desenvolvidos por outros produtores.
- Essa expansão levou à necessidade de ser criado e adaptado um padrão para a linguagem.
- Esta tarefa foi realizada pela American National Standards Institute (ANSI) em 1986 e ISO em 1987.

A linguagem SQL: história

- A SQL foi revista em 1992 e a esta versão foi dado o nome de SQL-92 ou SQL2.
- Foi revisto novamente em 1999 e 2003 para se tornar SQL:1999 (SQL3) e SQL:2003, respectivamente.
- A linguagem SQL, embora padronizado pela ANSI e ISO, possui muitas variações e extensões produzidas pelos diferentes fabricantes de sistemas gerenciadores de bases de dados.
- A linguagem pode ser migrada de plataforma para plataforma sem grandes mudanças estruturais.

A linguagem SQL: estrutura

- Linguagem de Definição de dados (DDL)
  - Subconjunto de comandos para definição e modificação de esquemas de relação (tabelas), remoção de tabelas, etc.
- Linguagem de Manipulação de dados (DML)
  - Subconjunto de comandos para inserir, remover e modificar informações em um banco de dados.
- Linguagem de Controle de Dados (DCL)
  - Subconjunto de comandos para controlar aspectos de autorização de dados e licenças de usuários;



Linguagem de Definição de Dados DDL

Linguagem de Definição de Dados:
Relação

- Relação
  - É a "matéria prima" para a construção de toda a teoria do modelo relacional e, por conseqüência, é o alicerce teórico de todo sistema de banco de dados baseado no modelo relacional.
  - Nos sistemas de banco de dados relacionais os dados são agrupados em TABELAS.
  - Uma tabela possui um nome e é constituída de uma ou mais colunas (ou campos). Os campos devem também possuir um nome, juntamente com o tipo de dado que será armazenado na coluna.

Linguagem de Definição de Dados:
Relação

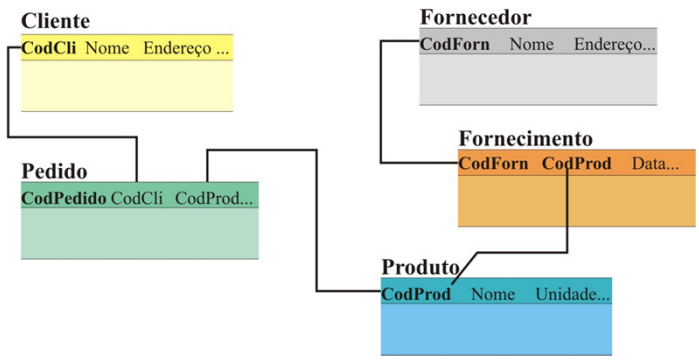
Cliente ⇒ Relação ou tabela

CodCli	Nome	Endereco
123	João	Rua Pio XI
567	Maria	Rua S. Francisco
678	**Joana**	**Av. Liberdade**
876	Gabriela	Av. Jatiúca
976	Ana Júlia	Av. São Paulo

⇒ coluna, campo ou atributo

⇒ linha ou registro

Linguagem de Definição de Dados



Linguagem de Definição de Dados: comandos

- CREATE objeto
  - cria um objeto (uma Tabela, por exemplo) no banco de dados.
- DROP objeto
  - Apaga/exclui um objeto do banco de dados.
- ALTER objeto
  - Altera a estrutura ou a configuração de um objeto no banco de dados

Linguagem de Definição de Dados:
tipos de dados (*Interbase*)

CHAR(n) **CHARACTER(n)**	Armazena caracteres alfanuméricos de tamanho fixo n. n = 1 a 32767
VARCHAR(n)	Cadeia de caracteres de comprimento variável e tamanho máximo de n caracteres. n = 1 a 32767
INTEGER	Dado numérico inteiro de tamanho fixo (32 bits). Representa valores no intervalo de: 2.147.483.648 a -2.147.483.647
SMALLINT	Representa valores inteiros de 16 bits no intervalo de: -37.768 a 32.767
NUMERIC(n,m) **DECIMAL(n,m)**	Dado numérico de tamanho variável, sendo n o número total de dígitos e m o número de casas decimais. O Parâmetro m é opcional
FLOAT	Dado numérico de ponto flutuante com precisão de 7 dígitos. Tem tamanho de 32 bits e armazena valores no intervalo de: 1.175×10^{-38} a 3.402×10^{38}
DATE	Data de tamanho fixo.
TIME	Hora de tamanho fixo
TIMESTAMP	Integra informações de data e hora
BLOB	*Binary Large Object*. Possui tamanho variável e permite armazenar dados, tais como imagens, audio, vídeo, etc. Os subtipos definem o conteúdo do campo. Os subtipos 0 e 1 são mais utilizados: 0 = dados binários de tipo indeterminado; 1 = Texto

Linguagem de Definição de Dados:
criando tabelas

Aluno

RA	numeric(8)
Nome	char(40)
RG	numeric(10)
Endereco	varchar(50)
Sexo	char(1)
dt_nasc	date

```
create table Aluno
  ( RA         numeric(8),
    nome       char(40),
    rg         numeric(10),
    endereco   varchar(50),
    sexo       char(1),
    dt_nasc    date
  )
```

Linguagem de Definição de Dados:
restrição de integridade

- Chave primária
  - A função da chave primária é identificar univocamente cada registro da tabela. Toda tabela deve possuir uma chave primária, que deve ser composta por um ou mais campos. Todo campo que compõe a chave primária dever ter a cláusula NOT NULL.

```
create table Aluno
  ( ra         numeric(8) not null primary key,
    nome       char(40),
    rg         numeric(10),
    endereco   varchar(50),
    sexo       char(1),
    dt_nasc    date
   )
```

Linguagem de Definição de Dados:
restrição de integridade

- Evitando valores nulos
  - É muito comum definirmos campos que não podem conter valores nulos. Isto é, o preenchimento do campo é obrigatório.
  - Para evitar que em algum momento um campo de uma tabela possa conter valor nulo (*null*) deve-se utilizar a cláusula **NOT NULL** após a definição do campo.

```
create table Aluno
  ( ra         numeric(8) not null primary key,
    nome       char(40)   not null,
    rg         numeric(10),
    endereco   varchar(50),
    sexo       char(1),
    dt_nasc    date
   )
```

Linguagem de Definição de Dados:
restrição de integridade

- Evitando valores inválidos
  - Existem situações onde um campo pode receber apenas alguns determinados valores. Para que o valor de um campo fique restrito a um determinado conjunto de valores, utiliza-se a cláusula **CHECK**.

```
create table Aluno
  ( ra          numeric(8)    not null primary key,
    nome        char(40)      not null,
    rg          numeric(8),
    endereco    varchar(50),
    sexo        char(1)       CHECK(sexo='M' or sexo='F'),
    dt_nasc     date
   )
```

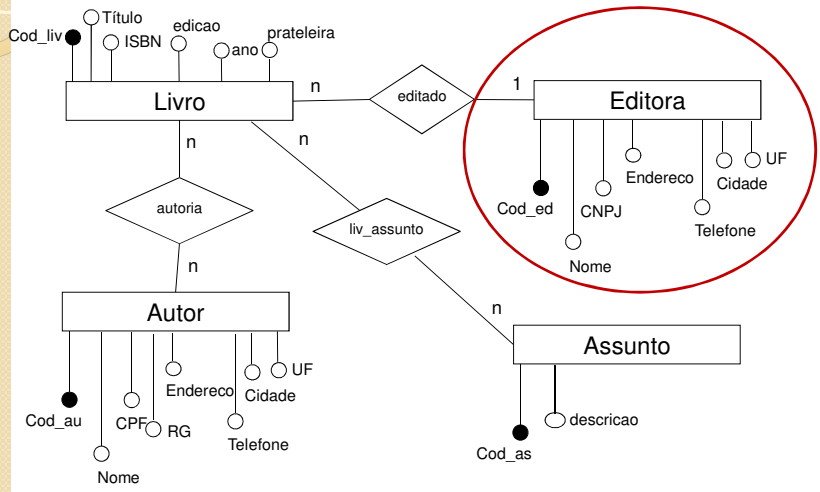
Linguagem de Definição de Dados:
restrição de integridade

- Evitando valores duplicados
  - Existem situações nas quais não deve existir dois iguais armazenados em uma mesma coluna. Isto é, valores inseridos em uma ou mais colunas são únicos para cada linha da tabela;
  - Para evitar que um valor armazenado em uma coluna de uma linha seja igual ao valor armazenado na mesma coluna de outra linha, utiliza-se a cláusula UNIQUE. A cláusula UNIQUE deve ser usada juntamente com a cláusula NOT NULL

```
create table Aluno
  ( ra          numeric(8)    not null primary key,
    nome        char(40)      not null,
    rg          numeric(8)    not null UNIQUE,
    endereco    varchar(50),
    sexo        char(1)       CHECK(sexo='M' or sexo='F'),
    dt_nasc     date
   )
```

Exercícios

Exercício 1



Exercício 1: resposta

```
create table editora
  ( cod_ed     numeric(3)  not null primary key,
    nome       varchar(40) not null,
    cnpj       numeric(15) not null UNIQUE,
    endereco   varchar(50),
    telefone   char(15),
    cidade     varchar(30),
    uf         char(2)
   )
```

Editora

Cod_ed	nome	CNPJ	endereco	telefone	cidade	UF

Linguagem de Definição de Dados:
integridade referencial

- É utilizada para garantir a Integridade dos dados entre as tabelas;

Aluno

RA	numeric(8)
Nome	char(40)
RG	numeric(10)
Endereco	varchar(50)
Sexo	char(1)
dt_nasc	date
cd_curso	integer

Curso

# cd_curso	integer
Nome	char(40)

Linguagem de Definição de Dados:
integridade referencial

1.

```
create table Curso
 ( cd_curso  integer  not null primary key,
   nome      char(40) not null,
 )
```

2.

```
create table Aluno
  ( ra         numeric(8) not null primary key,
    nome       char(40)   not null,
    rg         numeric(8) not null UNIQUE,
    endereco   varchar(50),
    sexo       char(1)    CHECK(sexo='M' or sexo='F'),
    dt_nasc    date,
    cd_curso   integer    references curso(cd_curso)
  )
```

Linguagem de Definição de Dados:
integridade referencial

Curso

cd_curso	nome
01	Ciência da Computação
02	Ciência da Informação

O campo **cd_curso** da tabela **Aluno** é chamado de **chave estrangeira** *(Foreign Key)*

Aluno

RA	nome	Rg	endereco	sexo	Dt_nasc	Cd_curso
1242532	Manoel	13243647	Rua Cinco	M	30/01/1963	01
1425534	Johanna	62736432	Rua São Paulo	F	14/11/1950	02
1565243	Maria	6152632	Rua Pio XII	F	15/09/1980	02
4537642	João	746732	Rua Leão 23	M	14/08/1970	01

Linguagem de Definição de Dados:
integridade referencial

Curso

cd_curso	nome
01	Ciência da Computação
02	Ciência da Informação

Mas... e se os dados da tabela "Curso" forem alterados ou excluídos ?

Aluno

RA	nome	Rg	endereco	sexo	Dt_nasc	Cd_curso
1242532	Manoel	13243647	Rua Cinco	M	30/01/1963	01
1425534	Johanna	62736432	Rua São Paulo	F	14/11/1950	02
1565243	Maria	6152632	Rua Pio XII	F	15/09/1980	02
4537642	João	746732	Rua Leão 23	M	14/08/1970	01

Linguagem de Definição de Dados:
integridade referencial

- Cláusulas complementares à cláusula REFERENCES

campo **REFERENCES** *outra_tabela* (*outro_campo*)
ON DELETE { **CASCADE** | **SET NULL** }
ON UPDATE { **CASCADE** | **SET NULL** }

Linguagem de Definição de Dados:
integridade referencial

```
create table Curso
  ( cd_curso  integer  not null primary key,
    nome      char(40) not null,
  )
```

```
create table Aluno
  ( ra         numeric(8) not null primary key,
    nome       char(40)   not null,
    rg         numeric(10) not null UNIQUE,
    endereco   varchar(50),
    sexo       char(1)    CHECK(sexo='M' or sexo='F'),
    dt_nasc    date,
    cd_curso   integer
               REFERENCES curso(cd_curso) ON DELETE CASCADE
  )
```

Linguagem de Definição de Dados:
integridade referencial

Curso

cd_curso	nome
01	Ciência da Computação
02	Ciência da Informação

Aluno

RA	nome	Rg	endereco	sexo	Dt_nasc	cd_curso
1242532	Manoel	13243647	Rua Cinco	M	30/01/1963	01
1425534	Johanna	62736432	Rua São Paulo	F	14/11/1950	02
1565243	Maria	6152632	Rua Pio XII	F	15/09/1980	02
4537642	João	746732	Rua Leão 23	M	14/08/1970	01

```
cd_curso REFERENCES curso(cd_curso) ON DELETE CASCADE
```

Linguagem de Definição de Dados:
integridade referencial

```
create table Curso
  ( cd_curso  integer  not null primary key,
    nome      char(40) not null,
  )
```

```
create table Aluno
  ( ra        numeric(8) not null primary key,
    nome      char(40)   not null,
    rg        numeric(10) not null UNIQUE,
    endereco  varchar(50),
    sexo      char(1)    CHECK(sexo='M' or sexo='F'),
    dt_nasc   date,
    cd_curso  integer
              REFERENCES curso(cd_curso) ON DELETE SET NULL
  )
```

Linguagem de Definição de Dados:
integridade referencial

Curso

cd_curso	nome
01	Ciência da Computação
02	Ciência da Informação

Aluno

RA	nome	Rg	endereco	sexo	Dt_nasc	cd_curso
1242532	Manoel	13243647	Rua Cinco	M	30/01/1963	01
1425534	Johanna	62736432	Rua São Paulo	F	14/11/1950	02
1565243	Maria	6152632	Rua Pio XII	F	15/09/1980	02
4537642	João	746732	Rua Leão 23	M	14/08/1970	01

```
cd_curso REFERENCES curso(cd_curso) ON DELETE SET NULL
```

Linguagem de Definição de Dados:
integridade referencial

```
create table Curso
  ( cd_curso  integer  not null primary key,
    nome      char(40) not null,
  )
```

```
create table Aluno
  ( ra         numeric(8) not null primary key,
    nome       char(40)   not null,
    rg         numeric(10) not null UNIQUE,
    endereco   varchar(50),
    sexo       char(1)    CHECK(sexo='M' or sexo='F'),
    dt_nasc    date,
    cd_curso   integer
               REFERENCES curso(cd_curso) ON UPDADE CASCADE
  )
```

Linguagem de Definição de Dados:
integridade referencial

01

Curso

cd_curso	nome
17	Ciência da Computação
02	Ciência da Informação

Aluno

RA	nome	Rg	endereco	sexo	Dt_nasc	cd_curso
1242532	Manoel	13243647	Rua Cinco	M	30/01/1963	17
1425534	Johanna	62736432	Rua São Paulo	F	14/11/1950	02
1565243	Maria	6152632	Rua Pio XII	F	15/09/1980	02
4537642	João	746732	Rua Leão 23	M	14/08/1970	17

```
cd_curso REFERENCES curso(cd_curso) ON UPDATE CASCADE
```

Linguagem de Definição de Dados:
integridade referencial

```
create table Curso
   ( cd_curso   integer   not null primary key,
     nome       char(40)  not null,
   )
```

```
create table Aluno
  ( ra          numeric(8)  not null primary key,
    nome        char(40)    not null,
    rg          numeric(10) not null UNIQUE,
    endereco    varchar(50),
    sexo        char(1)     CHECK(sexo='M' or sexo='F'),
    dt_nasc     date,
    cd_curso    integer
                REFERENCES curso(cd_curso) ON UPDADE SET NULL
   )
```

Linguagem de Definição de Dados

Integridade
Referencial

Curso

cd_curso	nome
17	Ciência da Computação
02	Ciência da Informação

01

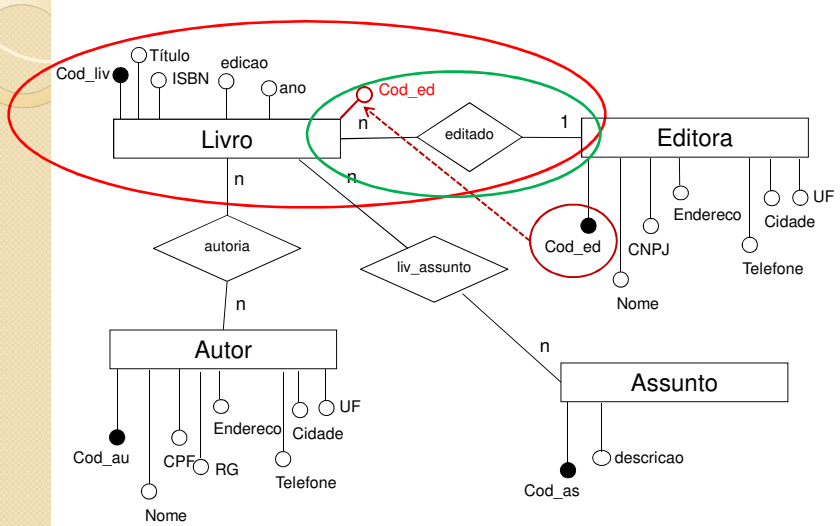
Aluno

RA	nome	Rg	endereco	sexo	Dt_nasc	cd_curso
1242532	Manoel	13243647	Rua Cinco	M	30/01/1963	~~01~~
1425534	Johanna	62736432	Rua São Paulo	F	14/11/1950	02
1565243	Maria	6152632	Rua Pio XII	F	15/09/1980	02
4537642	João	746732	Rua Leão 23	M	14/08/1970	~~01~~

```
cd_curso REFERENCES curso(cd_curso) ON UPDATE SET NULL
```

Exercícios

Exercício 2



Exercício 2: resposta

```
create table livro
  ( cod_liv      numeric(5)    not null primary key,
    titulo       varchar(40)   not null,
    isbn         numeric(15),
    edicao       numeric(2),
    ano          numeric(4),
    cod_ed       numeric(3)  references editora(cod_ed)
                             on update cascade
                             on delete set null
  )
```

Exercício 2: resposta

Editora

Cod_ed	nome	CNPJ	endereco	telefone	cidade	UF

Livro

Cod_liv	titulo	isbn	edicao	ano	Cod_ed

Linguagem de Definição de Dados:
alterando a estrutura de uma tabela

- Adicionando um novo campo

```
ALTER TABLE tabela
   ADD nome_campo tipo_dado
```

- Excluindo um campo

```
ALTER TABLE tabela
   DROP nome_campo
```

- Excluindo uma tabela

```
DROP TABLE tabela
```

- Alteração do nome do campo

```
ALTER TABLE tabela
   ALTER nome_campo TO novo_nome_campo
```

- Alteração do tipo (de dado) de um campo

```
ALTER TABLE tabela
   ALTER nome_campo TYPE novo_tipo
```

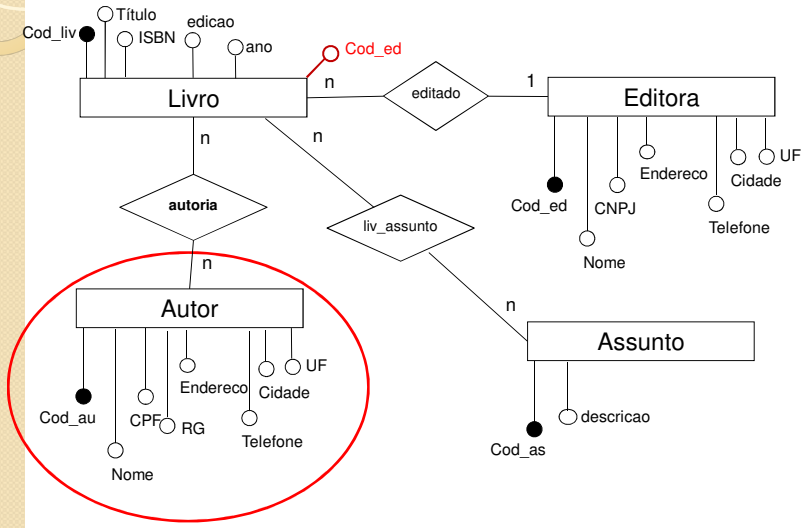
- Alterando a posição de um campo na tabela

```
ALTER TABLE tabela
   ALTER nome_campo POSITION n
```

- ***n*** é a nova posição do campo na tabela
- A posição do primeiro campo é zero

Exercícios

Exercício 3



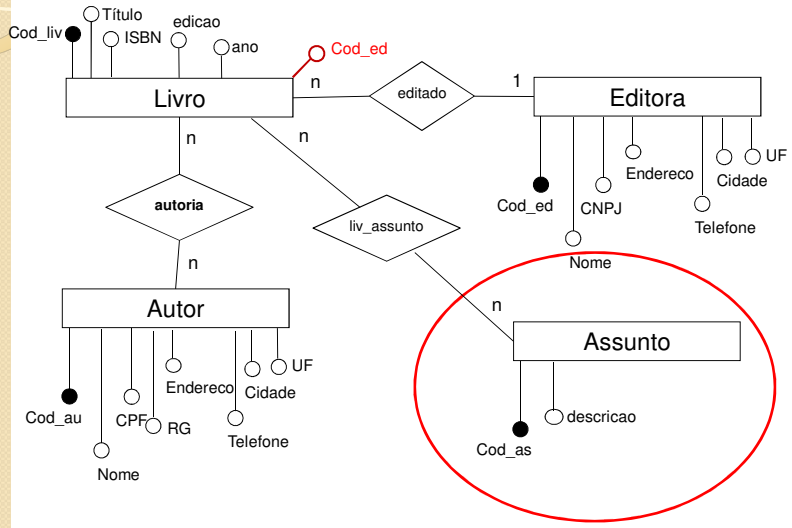
Exercício 3: resposta

```
create table autor
  ( cod_au       numeric(3)    not null primary key,
    nome         varchar(40)   not null,
    cpf          numeric(11),
    rg           numeric(10),
    endereco     varchar(50),
    telefone     char(15),
    cidade       varchar(30),
    uf           char(2)
   )
```

Autor

Cod_au	nome	cpf	rg	endereco	telefone	cidade	uf

Exercício 4



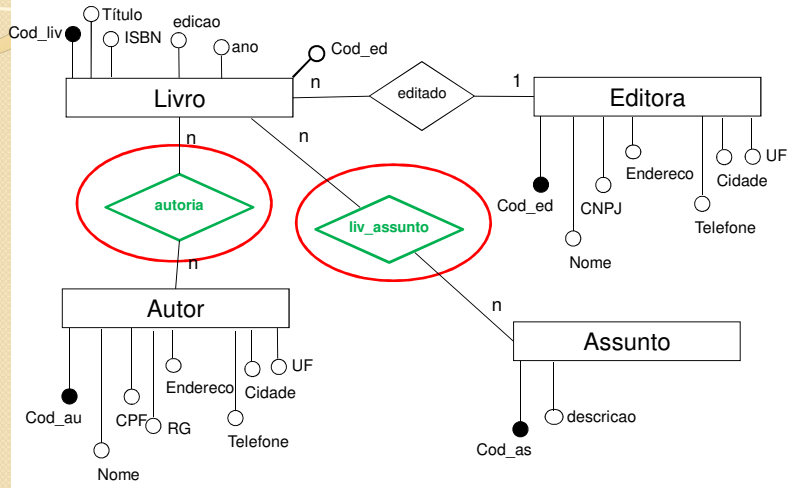
Exercício 4: resposta

```
create table assunto
  ( cod_as      numeric(3)    not null primary key,
    descricao   varchar(40)   not null,
  )
```

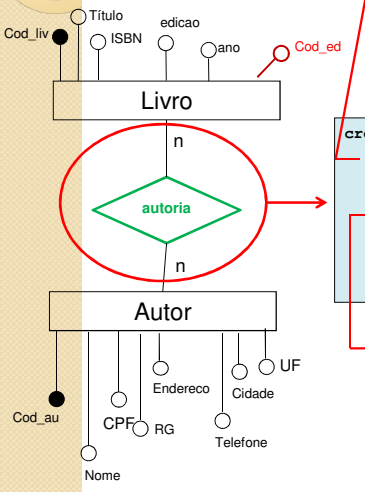
Assunto

Cod_as	descricao

Exercício 5:
transformar relacionamentos n:n em tabelas



Exercício 5

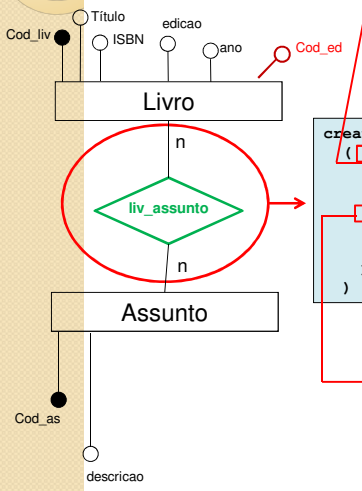



```
create table livro
 ( cod_liv      numeric(5)   not null primary key,
   titulo       varchar(40)  not null,
   isbn         numeric(15),
   edicao       numeric(2),
   ano          numeric(4),
   prateleira   integer,
   cod_ed       numeric(3)  references editora(cod_ed)
                            on update cascade
                            on delete set null
 )
```

```
create table autoria
 ( cod_liv   numeric(5) not null references livro(cod_liv)
                        on update cascade
                        on delete cascade,
   cod_au    numeric(3) not null references autor(cod_au)
                        on update cascade
                        on delete cascade,
   primary key(cod_liv, cod_au)
 )
```

```
create table autor
 ( cod_au      numeric(3)   not null primary key,
   nome        varchar(40)  not null,
   cpf         numeric(11),
   rg          numeric(10),
   endereco    varchar(50),
   telefone    char(15),
   cidade      varchar(30),
   uf          char(2)
 )
```

Exercício 5

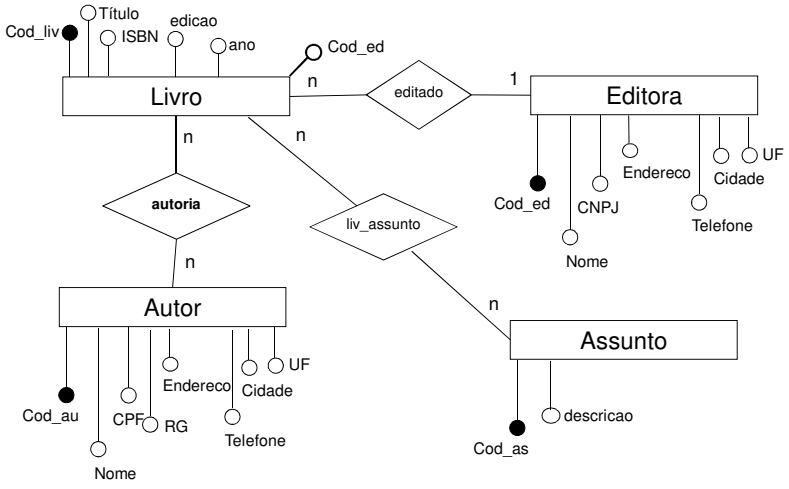


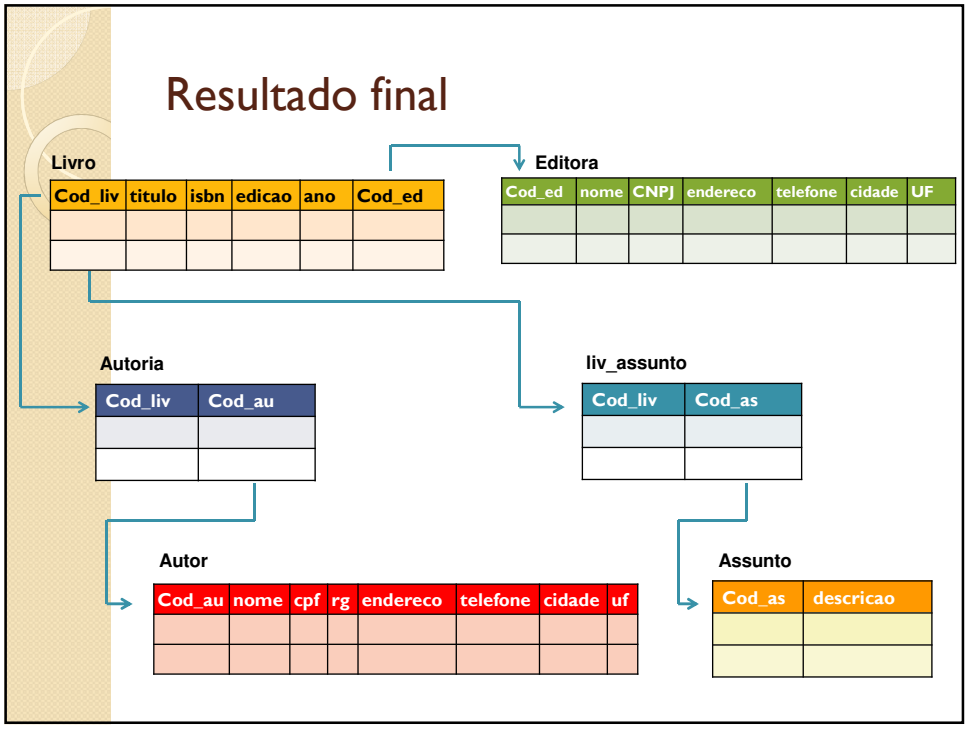

```
create table livro
( cod_liv      numeric(5)    not null primary key,
  titulo       varchar(40)   not null,
  isbn         numeric(15),
  edicao       numeric(2),
  ano          numeric(4),
  prateleira   integer,
  cod_ed       numeric(3)  references editora(cod_ed)
                           on update cascade
                           on delete set null
)
```

```
create table liv_assunto
  ( cod_liv    numeric(5) not null references livro(cod_liv)
                          on update cascade
                          on delete cascade,
    cod_as     numeric(3) not null references assunto(cod_as)
                          on update cascade
                          on delete cascade,
    primary key(cod_liv, cod_as)
  )
```

```
create table asunto
( cod_as       numeric(3)    not null primary key,
  nome         varchar(40)   not null,
)
```

Resultado final






```
create table editora
  ( cod_ed     numeric(3)  not null primary key,
    nome       varchar(40) not null,
    cnpj       numeric(15) not null UNIQUE,
    endereco   varchar(50),
    telefone   char(15),
    cidade     varchar(30),
    uf         char(2)
  )
```

```
create table livro
  ( cod_liv     numeric(5)   not null primary key,
    titulo      varchar(40)  not null,
    isbn        numeric(15),
      edicao      numeric(2),
      ano         numeric(4),
    cod_ed      numeric(3) references editora(cod_ed)
                on update cascade
                on delete set null )
```

```
create table autor
  ( cod_au      numeric(3)   not null primary key,
    nome        varchar(40)  not null,
    cpf         numeric(11),
    rg          numeric(10),
    endereco    varchar(50),
    telefone    char(15),
    cidade      varchar(30),
    uf          char(2) )
```

```
create table assunto
  ( cod_as      numeric(3)  not null primary key,
    descricao   varchar(40) not null,
  )
```

```
create table autoria
  ( cod_liv    numeric(5) not null reference
                                  on update
                                  on delete
    cod_au     numeric(3) not null reference
                                  on update
                                  on delete
    primary key(cod_liv, cod_au)
  )
```

```
create table liv_assunto
  ( cod_liv    numeric(5) not null references livro(cod_liv)
                                  on update cascade
                                  on delete cascade,
    cod_as     numeric(3) not null references assunto(cod_as)
                                  on update cascade
                                  on delete cascade,
    primary key(cod_liv, cod_as)
  )
```