Residência de Software

Banco de Dados - Modelo Relacional e comandos DDL

Modelo Relacional

- Modelo de dados para um SGBD, que se baseia no princípio que todos os dados estão guardados em tabelas (entidades do MER).
- Exemplo : Tabela Aluno

Num_Matrícula	Nome_Aluno	Data_Nascimento
1	Raul	1993-11-01
2	Clara	2000-09-17
3	Denis	1995-07-25
4	Fred	1996-06-13

- Regra : Nome de tabelas devem ser únicos no BD;

Modelo Relacional - Colunas || Atributos

- Um nome de atributo deve ser único em uma tabela e dizer exatamente o tipo de informação que ele representa
- Nome de uma coluna deve expressar exatamente o que ela armazena
- Sempre que possível utilizar prefixos padronizados

Num_Matricula	Nome do Aluno	data de nasc.

Num_Matricula	Nome_Aluno	Data_Nascimento

Modelo Relacional - Linha || Registros

- A tabela Aluno possui quatro registros
- Cada linha da tabela é única e possui um atributo identificador (Num_Matrícula)
  - Este atributo identificador é chamado de **chave_primária**, lembram?
- Regras :
  - Em uma tabela não devem existir linhas duplicadas;
  - As linhas de um tabela não seguem uma ordem específica.

1	Raul	01/11/1993
2	Clara	17/09/2000

1	Raul	01/11/1993
1	Clara	17/09/2000

Modelo Relacional - Domínio

- A tabela Aluno possui três atributos
- Para cada atributo existe um conjunto de valores permitidos chamado domínio daquele atributo
  - Para o atributo Num_Matricula o domínio é o conjunto de números naturais
  - Para o atributo Nome_Aluno o domínio é qualquer nome válido
  - Enquanto que para Data_Nascimento o domínio são as Datas no formato YYYY-MM-DD

Num_Matricula	Nome_Aluno	Data_Nascimento

Modelo Relacional - Tabelas e Entidades

- Para a criação de banco de dados, tabelas e atributos em um SGBD, utilizaremos a linguagem SQL que é composta de comandos de manipulação, definição e controle de dados
- Esses conjuntos de comandos de definição de dados são denominados pela sigla DDL (Data Definition Language), que disponibiliza um conjunto de comandos para criação (CREATE), alteração (ALTER) e remoção (DROP) de tabelas e outras estruturas

Comando CREATE DATABASE

- A maioria dos SGBDs disponibiliza ferramentas que permitem a criação de Banco de Dados, mas é possível criar o próprio Banco de Dados a partir de um comando SQL.
- A sintaxe do comando é:

```
CREATE DATABASE nome_do_banco_de_dados

CREATE DATABASE RESIDENCIA_DE_SOFTWARE
```

Comando DROP DATABASE

- O comando DROP DATABASE permite remover um determinado Banco de Dados, apagando todas as tabelas e estruturas associadas e, consequentemente, todos os dados existentes nelas
- A sintaxe do comando é:

```
DROP DATABASE nome_do_banco_de_dados

DROP DATABASE RESIDENCIA_DE_SOFTWARE
```

Comando CREATE TABLE

- O comando CREATE TABLE é o principal comando DDL da linguagem SQL. A criação de tabelas é realizada em SQL utilizando este comando.
- A sintaxe básica do comando é:

```
CREATE TABLE nome_da_table(Coluna1 Tipo, Coluna2 Tipo,
                           ColunaN Tipo)
```

```
CREATE TABLE Aluno(Num_Matricula INTEGER, Nome_Aluno
                   CHAR(50), Data_Nasc DATE )
```

Tipos de Dados

- Em SQL os tipos de dados são agrupados em 3 categorias:
  - Caracteres (Strings)
    - CHAR
    - CHAR(N)
    - VARCHAR(N)
  - Numéricos
    - NUMERIC
    - INTEGER ou INT
    - SMALLINT
    - FLOAT
    - BIT
  - Tempo e Data
    - DATE
    - TIME
    - DATETIME

Atributos NOT NULL

- No exemplo apresentado anteriormente

```
CREATE TABLE Aluno(Num_Matricula INTEGER, Nome_Aluno
CHAR(50), Data_Nasc DATE )
```

- Acima os valores podem ser nulos se nada for especificado. Para tornar um campo obrigatório podemos utilizar o atributo NOT NULL

```
CREATE TABLE Aluno(Num_Matricula INTEGER NOT NULL,
Nome_Aluno CHAR(50), Data_Nasc DATE )
```

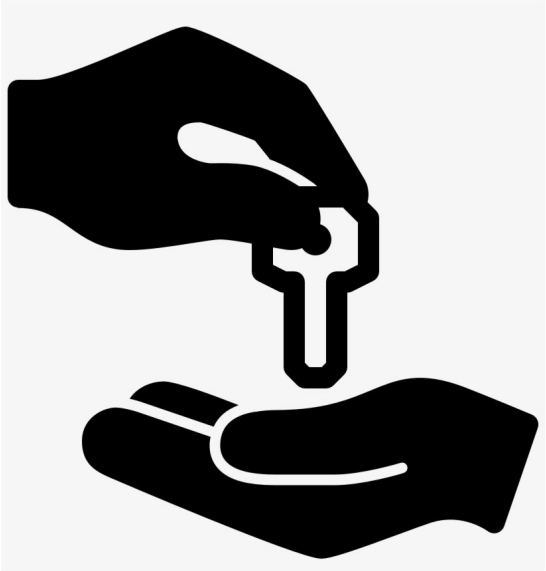
Outras restrições (*constraints*)

- Existem alguns outros tipos distintos de restrições que se podem aplicar a colunas:
  - NOT NULL
  - CHECK
  - UNIQUE
  - PRIMARY KEY
  - REFERENCES

Exercícios e Dinâmicas

Tipos de Chaves

- Chave Primária : chave que identifica cada tupla (linha)
- Chave Estrangeira : atributo ou conjunto de atributos de uma relação, que é a **chave primária** em **outra relação**



Chave Primária

- Uma chave primária(atributo identificador) é uma coluna ou um grupo de colunas que assegura a unicidade das linhas dentro de uma tabela;
- Chaves primárias são geralmente indicadas pela sigla PK (primary key);
- Exemplos de Chaves Primárias:
  - Produto (**Codigo_Produto**, Descrição, ...)
  - Veiculo (**Chassi**, Cor, Tipo, ...)
  - Usuario (**Login**,Senha, ...)
  - Conta_Bancaria(**Agencia**,**Conta_Corrente**, Titular)

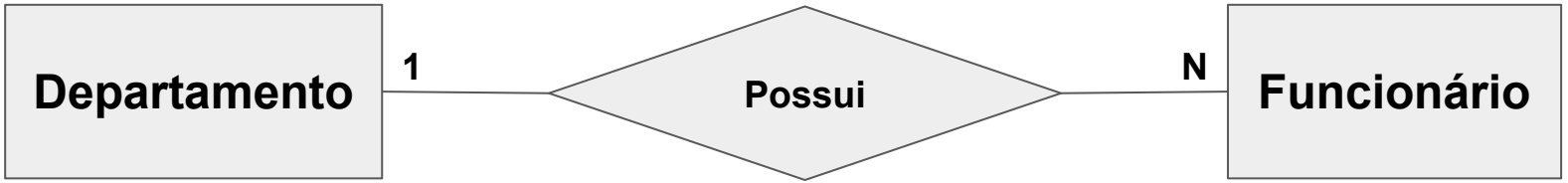
Obs: Os valores da chave primária não podem ser nulos e não podem existir duas chaves com o mesmo valor

Chaves Primárias - Recomendações

1. Selecione chaves primárias que permaneçam únicas;
2. Selecione chaves primárias que não tenham tendência a alterações;
3. Se possível, utilize chaves primárias simples;
4. Dê preferência a colunas numéricas para chaves primárias;
5. Selecione chaves primárias que sejam familiares;
6. Se não houver nenhuma coluna com chave primária candidata, utilize chaves primárias atribuídas pelo sistema (autonumeração)

Chave Estrangeira

- Em todo relacionamento entre entidades (MER) e entre tabelas (MR), deverá existir uma chave estrangeira que identifique a qual tupla ela está relacionada na tabela onde ela é a chave primária.
- No exemplo abaixo, na implementação para o modelo relacional, a tabela FUNCIONARIO deverá possuir uma chave estrangeira a qual DEPARTAMENTO ele está alocado. A chave estrangeira de FUNCIONÁRIO é uma chave primária na tabela DEPARTAMENTO.



Normalização

- Normalização é o processo que permite a simplificação da estrutura de um banco de dados, de modo que esta se apresente em um ótimo estado.
- A ideia é minimizar ou eliminar a redundância de informações. Apesar de existirem 6 formas normais, é considerado em um bom nível quando um esquema de banco de dados se encontra na terceira forma normal

1a Forma Normal

- Diz-se que uma relação está na primeira forma normal quando:
  - Não contém atributos multvalorados (grupo de atributos)
  - Não contém grupos repetititvos

Fatura (NumeroFatura (PK), CodigoCliente, NomeCliente, Endereco, CodigoProduto, DescricaoProduto, Preco, Quantidade)

1a Forma Normal

- Para fazer a passagem para a 1a FN será necessário separar a tabela em duas : **Fatura e Itens.**

Fatura (NumeroFatura (PK), CodigoCliente, NomeCliente, Endereco, CodigoProduto, DescricaoProduto, Preco, Quantidade)

Fatura (NumeroFatura,CodigoCliente, NomeCliente, Logradouro, Bairro,Cidade, Cep)

Itens (NumeroFatura (PK), CodigoProduto(PK), DescricaoProduto,Preco, Quantidade)

2a Forma Normal

- Diz-se que uma relação está na segunda forma normal quando:
  - Está na 1FN
  - Todos os atributos não chave dependem da totalidade da chave.

Itens (NumeroFatura (PK), CodigoProduto(PK), DescricaoProduto,Preco, Quantidade)

2a Forma Normal

- A tabela Itens não se encontra na 2FN porque os atributos **DescricaoProduto** e **Preço** não dependem de **NumeroFatura**, mas só de **CodigoProduto.**

Itens (NumeroFatura (PK), CodigoProduto(PK), DescricaoProduto,Preco, Quantidade)

Itens (NumeroFatura (PK), CodigoProduto (PK), Quantidade)

Produto (CodigoProduto (PK), DescricaoProduto,Preco)

3a Forma Normal

- Uma relação está na 3a FN se:
  - Está na 2a FN
  - Todos os seus atributos não chave NÃO são identificados por outro também não chave

Fatura (NumeroFatura(PK),CodigoCliente, NomeCliente, Logradouro, Bairro,Cidade, Cep)

3a Forma Normal

- A tabela Fatura não se encontra na 3FN porque os atributos **NomeCliente, Logradouro, Bairro, Cidade** e **Cep** são identificados por **CodigoCliente** e não por **CodigoFatura.** Para colocar na 3FN deve-se separar a tabela Fatura em duas tabelas:

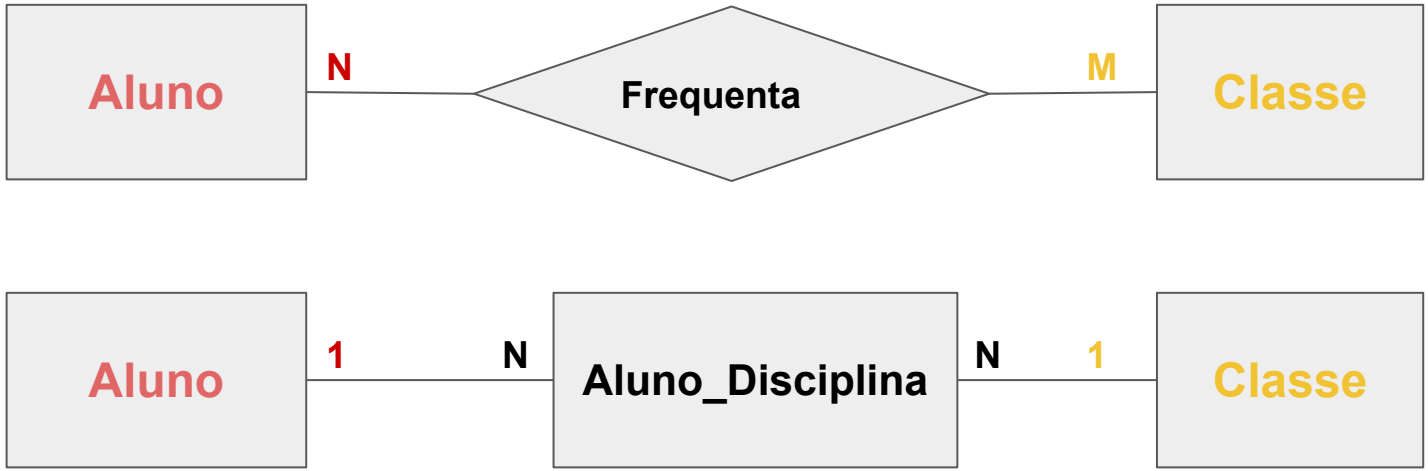
Fatura (NumeroFatura (PK), CodigoCliente, NomeCliente, Logradouro, Bairro,Cidade, Cep)

Fatura (NumeroFatura (PK), CodigoCliente)

Cliente (CodigoCliente (PK), NomeCliente, Logradouro, Bairro, Cidade, Cep)

Observação sobre relações N:M

Relacionamentos N:M vão sempre resultar na criação de uma nova entidade. O relacionamento N:M deixa de existir e passamos a ter dois relacionamentos 1:N.



Exercícios e Dinâmicas