

# BANCO DE DADOS I

V0

*Professor: Ivon Rodrigues Canedo*

## ÍNDICE

1.	Banco de Dados	003
1.1.	Um Definição	003
1.2.	Sistema Gerenciador de Banco de Dados	003
1.3.	Tipos de Modelos de Dados	003
2.	O Modelo de Dados	007
2.1.	Componentes de um Modelo de Dados	007
3.	Convenções para Construção de um DER	019
3.1.	Nomeclatura dos elementos de um DER	020
3.2.	Exemplos de DER	020
4.	Exercícios	021
5.	Normalização de Dados	022
6.	Modelo Relacional	031
7.	Restrições no Modelo Relacional	032
8.	Exercícios de Modelagem de Dados	037
9.	SQL	045
9.1.	Create Table	046
9.2.	Alter Table	049
9.3.	Drop Table	050
9.4.	Create Index	051
9.5.	Constraint	052
9.6.	Insert Into	054
9.7.	Delete	056
9.8.	Update	057
9.9.	Select Into	058
9.10.	Select	059
9.11.	Atributos ALL, DISTINCT, DISTINCTROW, TOP	060
9.12.	Clausula From	062
9.13.	Clausula Group BY	063
9.14.	Clausula Having	064
9.15.	Clausula In	065
9.16.	Clausula Order BY	066
9.17.	Clausula Where	068
9.18.	Funções Agregadas SQL	070
9.19.	Operação Inner Join	073
9.20.	Operação Left Join e Right Join	074
9.21.	Operação Union	075

9.22. Subconsulta Sql	076
9.23. Instrução Transform	078
9.24. Declaração Parameters	079
9.25. Operador Between...And	080
9.26. Operador In	081
9.27. Operador Like	081
9.28. Caracteres Curinga	082
10. Create User ou Group	083
11. Add User	084
12. Drop User ou Group	084
13. Alter User ou Database	085
14. Grant	085
15. Revoke	087
16. Tipos de Dados SQL	088
17. Exercícios SQL Resolvidos	091
18. O Banco Acadêmico	111
19. Exercícios SQL	137
19.1. Banco de Dados – Acadêmico – Parte I	138
19.2. Banco de Dados – Cirúrgico	143
19.3. Banco de Dados – Família Zacharias	145
19.4. Banco de Dados – Transporte	147
19.5. Banco de Dados – Acadêmico – Parte II	149
20. Exercícios Diversos	153

## **BANCO DE DADOS**

### **Uma Definição**

É um conjunto de dados, relativos a um determinado ambiente, por exemplo, um empresa de fornecimento de energia elétrica, armazenados em um ou vários computadores e que guardam entre si algum relacionamento.

### **Exemplo**

Banco de Dados de Recursos Humanos de uma empresa;

Banco de Dados de Aplicações Financeiras;

Banco de Dados de uma empresa de energia elétrica.

### **Base de Dados**

É o conjunto de todos os dados de um determinado ambiente, estejam eles armazenados em computador ou não.

### **Exemplo**

Banco de Dados de Recursos Humanos + as correspondências expedidas e recebidas, pela empresa, e que estão guardadas nos armários de cada departamento.

### **Sistema Gerenciador de Bancos de Dados (SGBD)**

É constituído por um conjunto de dados inter-relacionados e um conjunto de programas para acessá-los. Sua característica principal é prover uma maneira adequada de recuperação e armazenamento de dados, no Banco de Dados. Regra geral, um SGBD é projetado para gerenciar grandes volumes de dados.

### **Características de um SGBD**

- Gerenciar grandes volumes de dados
- Facilitar a eliminação de redundância e inconsistência de dados
- Facilitar o armazenamento e acesso aos dados
- Garantir o acesso a vários usuários ao mesmo tempo
- Garantir a segurança dos dados (Por exemplo, garantir a recuperação dos dados caso haja danificação do meio onde estão armazenados. Garantir segurança de acesso).
- Garantir a integridade dos dados

### **Abstração de Dados**

É a possibilidade de entender uma ambiente se preocupando apenas com seus aspectos mais importantes. No caso dos Bancos de Dados abre a possibilidade a seus usuários de poderem acessar aos dados sem a necessidade de se preocuparem com os detalhes de como os dados são armazenados.

### **Níveis de Abstração**

#### **Nível Físico**

Descreve como os dados são realmente armazenados. Neste nível de abstração trabalham os DBAs.

#### **Nível Conceitual**

Descreve quais dados estão armazenados e como eles se relacionam. Neste nível os usuários não se preocupam os aspectos físicos do armazenamento de dados. Aqui trabalham os DBAs e os Analistas de Aplicação.

### Nível de Visão

Uma visão descreve parte de um banco de dados, de modo que , usuários do banco tenham acesso apenas aos dados que lhes dizem respeito. Um banco de dados tem muitas visões.

### Modelos de Dados

Identificam os dados de um determinado ambiente, as relações entre eles e suas restrições de integridade.

### Tipos de Modelos de Dados

#### Modelo Lógicos Baseados em Objetos

##### Modelo de Entidade x Relacionamento

Consiste num conjunto de objetos representativos de uma ambiente chamados *entidades* e nos relacionamentos que mantêm entre si. Uma entidade se distingue de outras entidades pelos atributos que ela contém.

##### Representação gráfica de um modelo de entidade-relacionamento

Retângulos – Representam conjuntos de entidades

Losangos – Representam os relacionamentos

Linhas – Ligam os relacionamentos às entidades relacionadas

##### Modelo Orientado a Objeto

É baseado em objetos representativos de um ambiente que se relacionam entre si. Aqui os objetos contém segmentos de códigos, denominados *métodos*, que os manipulam. Um objeto só pode ser acessado através de seus métodos. Uma solicitação de acesso a um objeto é denominada *mensagem*

É um conjunto de objetos com os mesmos valores e os mesmos *métodos*.

#### Modelos Lógicos Baseados em Registros

Os modelos baseados em registros são usados nos níveis de abstração conceitual e visual.

##### Modelo Relacional

Representa os dados e seus relacionamentos através de tabelas. Cada tabela corresponde a um conjunto de entidades do modelo relacional e contém um número de colunas com nomes únicos, sendo que cada coluna representa um atributo da entidade. Os relacionamentos são representados por dados contidos dentro das próprias tabelas. Exemplo: Oracle

##### Modelo de Redes

Os dados são representados por uma coleção de registros e os relacionamentos entre os dados são representados por ponteiros. Os registros nos bancos de dados são organizados como coleções de grafos arbitrários. Exemplo: IDS-II da ABC-BULL.

##### Modelo Hierárquico

Os dados são representados por uma coleção de registros e os relacionamentos entre os dados são representados por ponteiros. Os dados são organizados em árvores.

**Instância de um Banco de Dados**

É o conjunto de informações do banco em um determinado momento

**Esquema**

É a representação do projeto do banco de dados. Não mudam com frequência. Em um banco de dados temos esquemas físicos e conceituais.

**Independência de Dados**

É possibilidade de mudar esquemas de um nível de abstração sem comprometer os esquemas de outros níveis

**Independência Física de Dados**

É a característica que os bancos de dados devem ter e que consiste na possibilidade de que alterações feitas no esquema físico não exigirem alterações nos programas escritos.

**Independência Lógica de Dados**

É uma característica dos bancos de dados que permite mudanças nos esquemas conceituais sem exigir mudanças nos programas escritos. É difícil de ser conseguida.

**Linguagens de Definição de Dados**

São aquelas usadas para definir o esquema de um banco de dados. São as DDLs. O resultado da execução de comandos das DDLs é um conjunto de tabelas que são armazenadas no dicionário de dados. São as DDLs que definem, por exemplo, os métodos de acesso de um banco de dados.

**Linguagens de Manipulação de Dados**

São as linguagens que manipulam informações de um banco de dados. São elas que fazem inclusões de novos dados, remoções e alterações de dados existentes e busca (recuperação) de dados armazenados. São as DMLs.

**DMLs Procedurais**

Exigem a especificação de como obter os dados.

**DMLs Não-Procedurais**

Não exigem a especificação de como obter os dados

**Gerenciador de Banco de Dados**

É constituído de um conjunto de programas que estabelecem a interface entre os dados armazenados e as solicitações ao sistema. Além disso, esses programas garantem um armazenamento de dados que facilita o desempenho do Banco, como por exemplo, um bom tempo de resposta para as solicitações às quais é submetido.

**Objetivos de um Gerenciador de Banco de Dados**

- Interagir com o gerenciador de arquivos. Geralmente esses gerenciadores de arquivos são fornecidos pelos sistemas operacionais. É de responsabilidade desses gerenciadores o atendimento à solicitação de dados.
- Garantir Integridade do Dados
- Garantir Segurança de Acesso

- Recuperação de Dados
- Controlar Concorrência de Acesso.

### **Administrador de Banco de Dados (DBA)**

É a pessoa que tem sob sua responsabilidade a gerência dos dados do banco e dos programas de acesso. É ele que garante a sua implantação e operação.

### **Principais Funções de um DBA**

Definir e alterar esquemas;  
Definir de estruturas de armazenamento e métodos de acesso aos dados;  
Conceder autorização de acesso;  
Especificar restrições de integridade.

### **Usuários de Banco de Dados (Proposto pelo prof. Ronaldo Lopes)**

#### **1. Projetistas**

- identificam: dados + restrições de integridade + requisitos de desempenho
- definem modelo conceitual e lógico da aplicação
- em algumas organizações podem definir modelo físico
- precisam conhecer o negócio

#### **2. Administradores de Banco de Dados (ABD ou DBA)**

- administram banco de dados
- definem critérios de acesso
- monitoram desempenho
- definem projeto físico
- definem estratégia de backup e recuperação de falhas
- precisam conhecer o SGBD

#### **3. Usuários Finais**

- casuais
- novatos (parametrizados)
- sofisticados

#### **4. Analistas e Programadores**

#### **5. Administradores de Dados (AD)**

#### **6. Projetistas e implementadores de SGBD**

#### **7. Desenvolvedores de ferramentas**

#### **8. Operadores e pessoal de manutenção**

## O MODELO DE DADOS

### Uma Definição

O Modelo de Dados é uma representação das necessidades de dados de um determinado ambiente e de como esses dados se relacionam. É uma das primeiras atividades que deve ser executada ao longo do processo de identificação e compreensão de um ambiente, tendo em vista necessidades de automatização. É um dos produtos da fase de Análise do Ciclo de Vida de um projeto de desenvolvimento de um sistema. Construir um Modelo de Dados significa: coletar e documentar informações relevantes do ambiente estudado; representar as informações, de forma clara e objetiva, e num formato padrão que possa facilitar o entendimento dos participantes do processo; definir, de maneira clara, o escopo do ambiente modelado; adquirir o entendimento do ambiente através de refinamentos sucessivos do modelo; e representar graficamente as necessidades de informação independentemente do Software e do Hardware a serem usados na implementação do Sistema.

### 1. Componentes de um Modelo de Dados

Um Modelo de Dados é composto de: entidade, tipo de entidade, atributo, relacionamentos e dicionário de dados.

#### 1.1. Entidade

Chamamos de entidade, qualquer coisa real ou abstrata, de um determinado ambiente, sobre a qual precisamos guardar informações. Se estamos modelando o ambiente de uma biblioteca, por exemplo, então as informações a respeito dos livros devem estar representadas pela entidade: Livro. As informações relativas aos usuários da biblioteca poderiam ser representada pela entidade: Cliente. As informações relativas ao empréstimo de livros seriam representadas pela entidade: Empréstimo. As reservas de livros por: Reserva, e assim por diante. Nesse ambiente as informações da entidade Livro poderiam ser: *nome do livro*, *ISBN do livro*, e *título do livro*. Para a entidade Cliente poderíamos ter: *cpf*, *nome*, *endereço* e *telefone*. Para a entidade Empréstimo: *data do empréstimo*, *data provável de devolução* e *taxa de multa para o caso de devolução com atraso*. A entidade Reserva poderia ter: *data da reserva*, e *data provável de disponibilidade do livro*.

É claro que as informações representadas por uma entidade dependem do ambiente onde ela está inserida. Por exemplo: uma pessoa para o Ministério da Fazenda é vista como um contribuinte de impostos. E dentro deste contexto as informações relevantes de pessoa seriam: *cpf*, *renda anual*, *despesas médicas*, *despesas com instrução*, etc. Já o Ministério da Educação poderia ter outras necessidades de informações sobre uma pessoa, como por exemplo: *cpf*, *nível de escolaridade*, *idade*, *data de nascimento*, etc. O Ministério da Saúde, certamente teria interesse em informações sobre saúde: *tipo sanguíneo*, *data de nascimento*, etc. Podemos concluir que, uma entidade só deve conter informações que dizem respeito, ou que são necessárias, ao ambiente que representa.

#### 1.2. Tipo de Entidade

Um tipo de entidade é definido como sendo o conjunto de todas entidades de uma mesma natureza, ou seja, que tenham as mesmas características. Por exemplo, o conjunto de todas as entidades Livros de uma biblioteca constituem o Tipo de Entidade LIVRO. Engenharia de Software, Análise Estruturada, Análise Orientada a Objeto, poderiam ser entidades do tipo de entidade LIVRO. O conjunto de todos os empréstimos feito pela biblioteca comporia o tipo de entidade EMPRÉSTIMO. O conjunto de todas as pessoas que usam a biblioteca representaria o tipo de entidade CLIENTE.

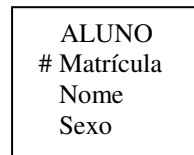
##### Tipo de Entidade Primária

É o tipo de entidade que existe por si mesma. Sua identificação completa é feita pelos seus próprios atributos.

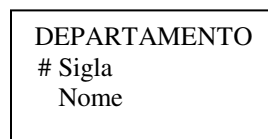


**Exemplos:**

1. A Entidade ALUNO é uma entidade primária porque é identificada pelos seus próprios atributos. O seu identificador pode ser o atributo *Matrícula* do aluno.



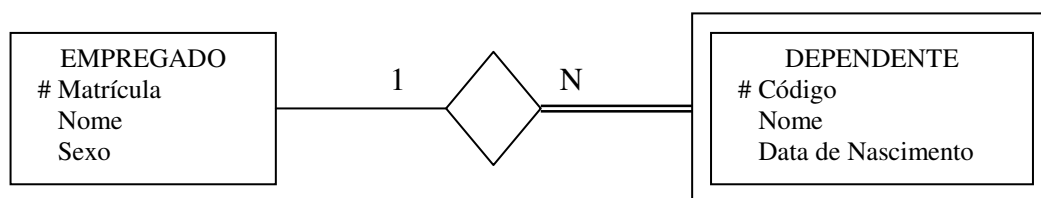
2. A Entidade DEPARTAMENTO é uma entidade primária pois tem como atributo identificador um atributo próprio. A sigla do departamento, por exemplo.

**Tipo de Entidade Fraca ou Dependente**

É a entidade cuja identificação não pode ser feita por seus próprios atributos. Para sua identificação completa precisamos de atributos de outra entidade.

**Exemplo**

1. A entidade DEPENDENTE é uma entidade fraca pois para a sua identificação há que se utilizar atributos da entidade EMPREGADO (A Matrícula do empregado, por exemplo). Ou seja, quando falamos, João Vieira (Dependente) precisamos dizer de quem ele é dependente (Empregado) para que se possa identificá-lo completamente. Assim: João Vieira é dependente de Francisco da Rocha (Empregado)

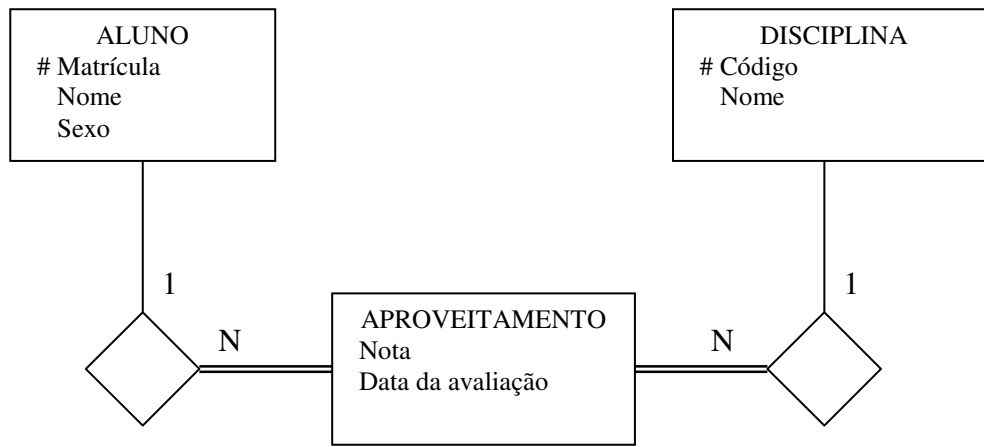
**Entidade Associativa**

É a entidade que não se identifica por si mesma e sua existência depende da existência de duas ou mais outras entidades. Compõem seu identificador, os identificadores das entidades que se associaram para lhe dar origem.

**Exemplo**

1. No diagrama abaixo a entidade, APROVEITAMENTO é uma entidade associativa porque a sua identificação só possível a partir da Matrícula, identificador da entidade ALUNO e de Código, identificador da entidade DISCIPLINA. Ou sejam, quando nos referimos ao aproveitamento 7,3, por

exemplo, ele só tem sentido quando associado a uma aluno e a uma disciplina. Assim: *João Ribeiro Ferraz* (Aluno) obteve a *nota 7,3* (Aproveitamento) em *Banco de Dados* (Disciplina).



### 1.3. Atributos

São partes específicas de uma determinada entidade. São as informações que caracterizam a entidade.

#### Exemplos de Atributos

Poderiam ser atributos de uma entidade Aluno: *nome, número da matrícula, cpf, data de ingresso no curso, endereço, telefone e data de nascimento*. Uma entidade Fornecedor poderia Ter como atributos: *Cgc, nome, Razão Social, Endereço, e Capital Social*. Cada entidade tem valores específicos para seus atributos que diferir ou ser iguais aos valores dos atributos de outras entidades de um mesmo tipo de entidade.

#### Valor de um Atributo

Chamamos valor de um atributo ao conteúdo que um atributo pode ter. *Marcos Ferreira, Rosa Cristina, Deusdete da Cunha* poderiam ser valores da entidade Aluno. *Casa do Barata, Mesbla, C&A* seriam valores do atributo nome da entidade Fornecedor.

#### Domínio de um Atributo

É o conjunto de valores que um atributo pode assumir. Exemplo: *Masculino, Feminino* são o domínio do atributo Sexo da entidade Aluno. O atributo Nota da entidade aluno tem o domínio: {números reais de 0 a 10}.

#### Tipos de Atributos de uma Entidade

##### Único

Cada entidade tem um valor diferente para este atributo. A matrícula de um aluno em um curso é um atributo único porque não existe outro aluno matriculado com o mesmo número de matrícula.

##### Não-Único

Quando o valor pode se repetir em várias entidades. Por exemplo, o aproveitamento de um aluno. Mais de um aluno pode ter a mesma nota.

**Obrigatório**

Quando tem que existir um valor para este atributo em toda entidade. Por exemplo, o nome do aluno na entidade ALUNO.

**Simples**

Quando possui um domínio simples. Por exemplo, o atributo sexo tem um domínio simples pois é formado pelo conjunto (único) das letras F e M.

**Composto**

Quando possui mais de um domínio simples. Endereço de uma pessoa, por exemplo. Ele é formado pelos domínios, simples, dos Logradouros, dos Bairros, das Cidades, dos Estados e dos CEP's.

**Univalorado**

Quando tem um único valor para cada entidade. Por exemplo, o número de matrícula de um aluno. Cada aluno tem um único número de matrícula.

**Multivalorado**

Quando pode ter mais de um valor para cada entidade. Por exemplo, o telefone de uma pessoa. Uma pessoa pode ter mais de um telefone. O do trabalho e da residência.

**Derivado**

Quando o seu conteúdo depende do conteúdos de outros atributos. Por exemplo, o total de uma nota fiscal é formado pela soma dos totais de cada item componente da nota fiscal.

**Não derivado**

Quando ele não pode ser obtido a partir de outros atributos. Por exemplo, nome de um aluno.

**Identificador**

É o atributo ou atributos que identificam uma entidade de um tipo de entidade de maneira única. Por exemplo a matrícula do estudante. Ou a matrícula do aluno e o código da disciplina no tipo de entidade APROVEITAMENTO.

**Não Identificador**

Quando o identificador não identifica por si só um entidade dentro de um tipo de entidades. Por exemplo, o nome do aluno não identifica o aluno dentro to tipo de entidade ALUNO.

**Matriz de Definição dos Atributos**

Atributos	ID	OB	MV	AD	AC	CE	NAT	TAM	DEC	DOM
Matrícula	S	S					N	4		> zero
Nome		S					C	40		<> Nulo
Telefone			S				C	14		<> Nulo

### 1.4. Relacionamentos

Chamamos de relacionamento a associação entre duas entidades ou entre uma entidade e ela mesma. Para expressar, em um modelo, quais as disciplinas nas quais um aluno está matriculado nós poderíamos definir o relacionamento: *O aluno está matriculado em...* O relacionamento para expressar os dependentes de um determinado empregado seria: *Empregado tem dependentes*.

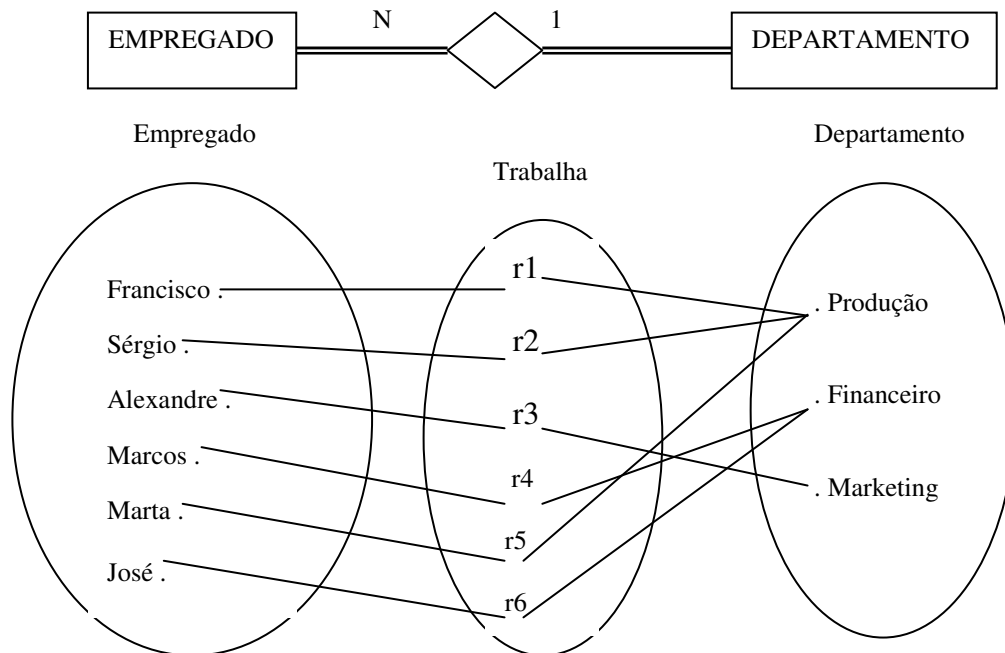
#### Cardinalidade de um relacionamento

Indica quantas entidades de um tipo de entidade participam de um relacionamento.

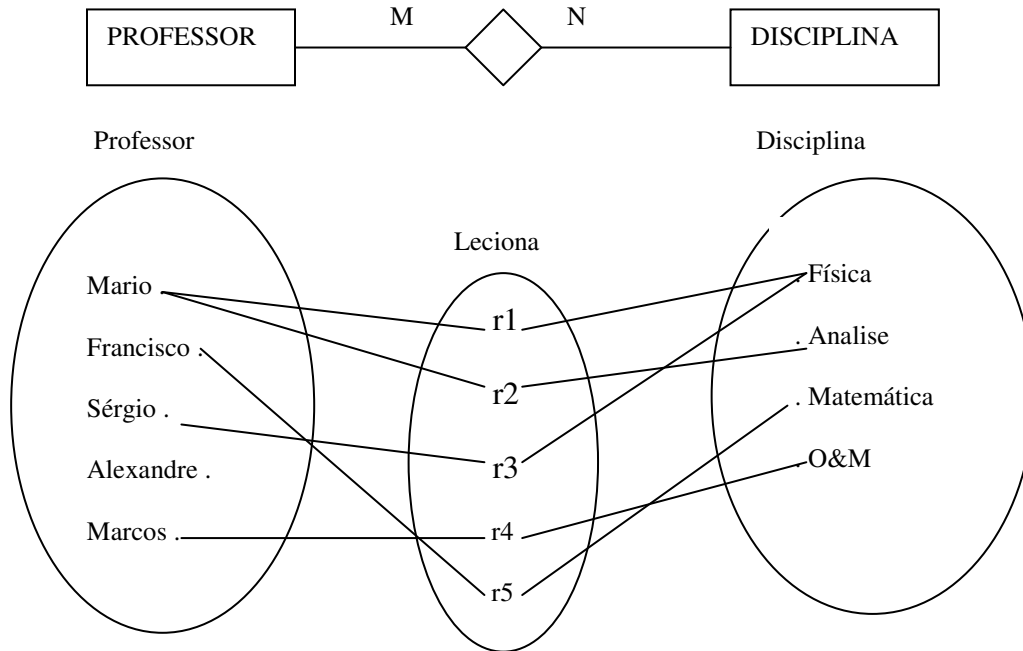
#### Restrições de relacionamento

Indica a participação ou não de uma entidade no relacionamento em causa.

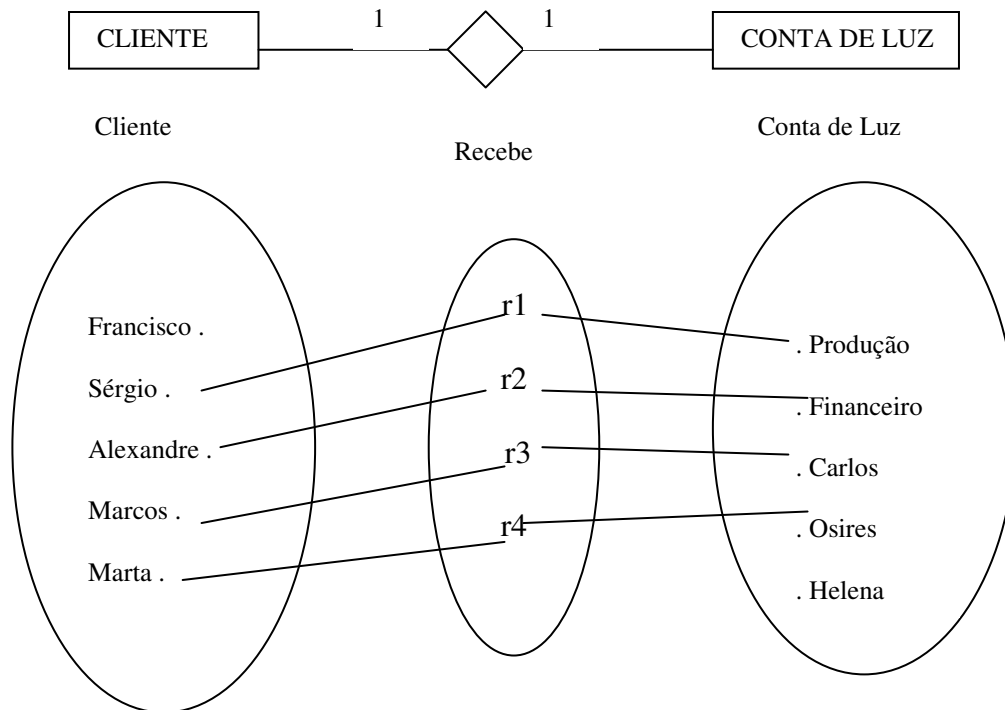
#### Uma Visão de um Relacionamento 1:N entre dois Tipos de Entidades



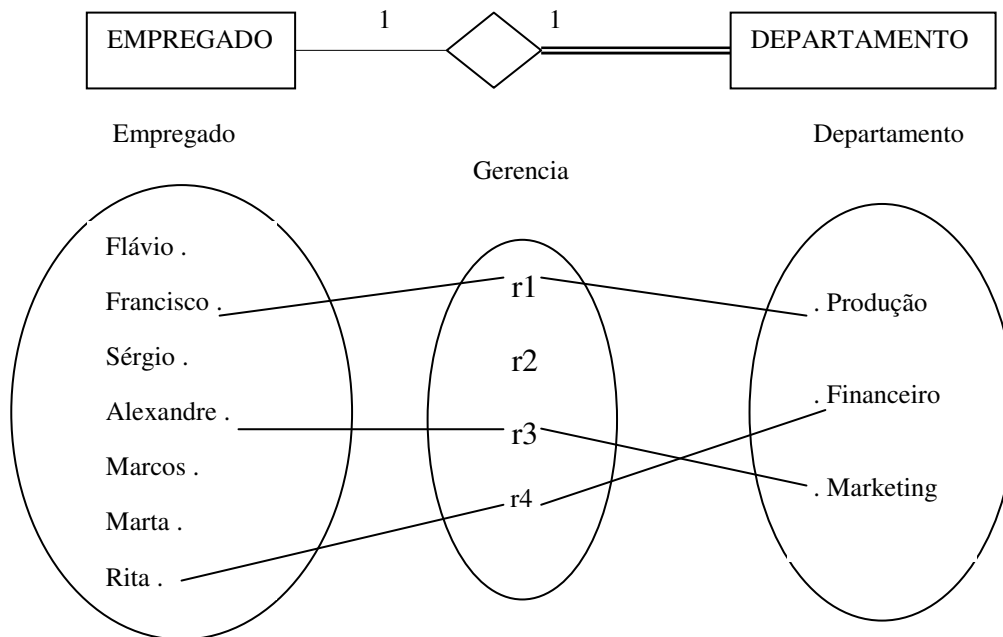
No Departamento de Produção (1) trabalham N (3) Empregados ( Francisco, Sérgio, Marta)  
 No Departamento Financeiro (1) trabalham N (2) Empregados ( Marcos, José)  
 No Departamento de Marketing (1) trabalha N (1) Empregado (Alexandre)

**Uma Visão de um Relacionamento M:N Entre dois Tipos de Entidades**

O professor Mário (1) leciona Física e Análise (2) e Física (1) é lecionada por Mário e Sérgio (2)  
O professor Francisco (1) leciona Matemática (1) e Matemática (1) é lecionada por Francisco (1)

**Uma Visão de um Relacionamento 1:1 Entre dois Tipos de Entidades**

**Uma Visão do Relacionamento 1:1 Entre dois Tipos de Entidades sem  
Obrigatoriedade do laod EMPREGADO**



Repare que existe empregados que não estão associados a departamento nenhum. São os empregados que não gerenciam departamentos.

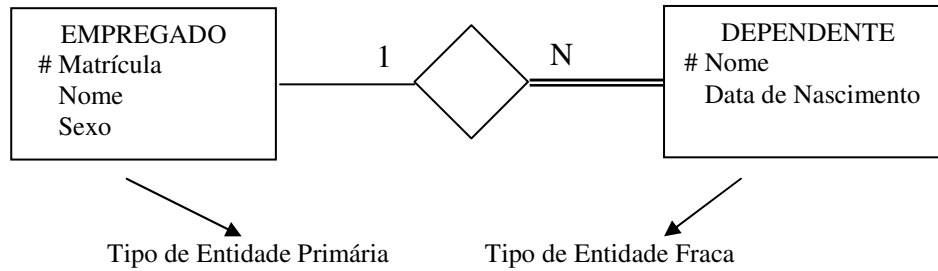


## Tipos de Relacionamentos

### Relacionamento tipo Dependência

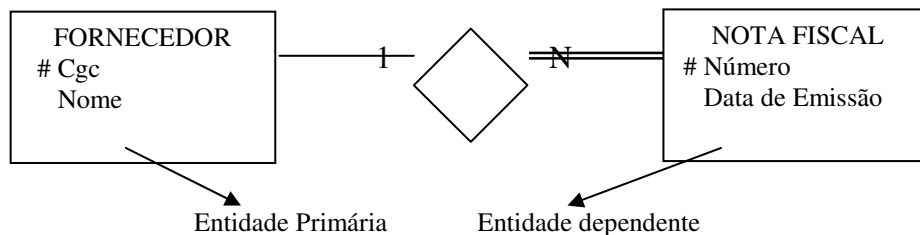
Chamamos de *Relacionamento de Dependência* ao relacionamento entre um tipo de entidade *primária* e um tipo de entidade dependente (fraca).

### Exemplo



O Tipo de entidade EMPREGADO compõe-se de entidades *primárias* porque essas entidades são identificadas completamente por seus atributos. É razoável imaginarmos que numa mesma empresa não exista empregados com a mesma matrícula. Já as entidades de DEPENDENTE são do tipo *fraca* porque os seus atributos não as identificam completamente. Suponhamos que *Pedro Rodrigues* seja filho de *Francisco Moreira*, cujo número de matrícula seja 10. Se falamos apenas *Pedro Rodrigues*, não o identificamos, porque pode existir outros *Pedro Rodrigues* filhos de outros empregado que não seja o *Francisco Moreira* de matrícula 10. Para que Pedro Rodrigues seja completamente identificado precisamos associá-lo ao empregado do qual ele é dependente. Então dizemos: *Pedro Rodrigues é dependente de Francisco Moreira de matrícula 10*

### Outro Exemplo

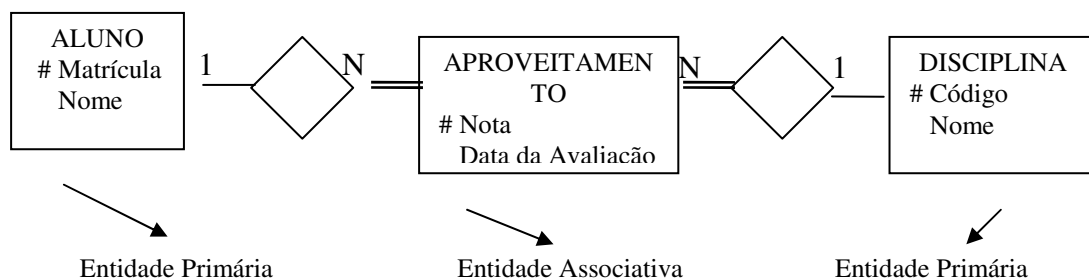


O tipo de entidade FORNECEDOR também é uma entidade primária porque é identificada completamente por seus atributos. Não existe dois fornecedores diferentes com o mesmo número de CGC. NOTA FISCAL é um tipo de entidade composto por entidades fracas ou dependentes, porque para identificar uma determinada nota fiscal completamente precisamos dizer de qual fornecedor é a nota fiscal, visto que podem existir notas fiscais de mesmo número pertencentes a fornecedores diferentes.

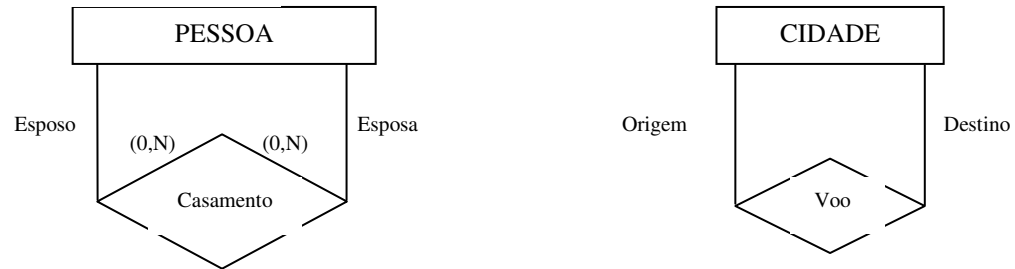
### Relacionamento tipo Associativo

Um relacionamento é do tipo associativo se ele relaciona uma entidade primária a uma entidade associativa.

### Exemplo

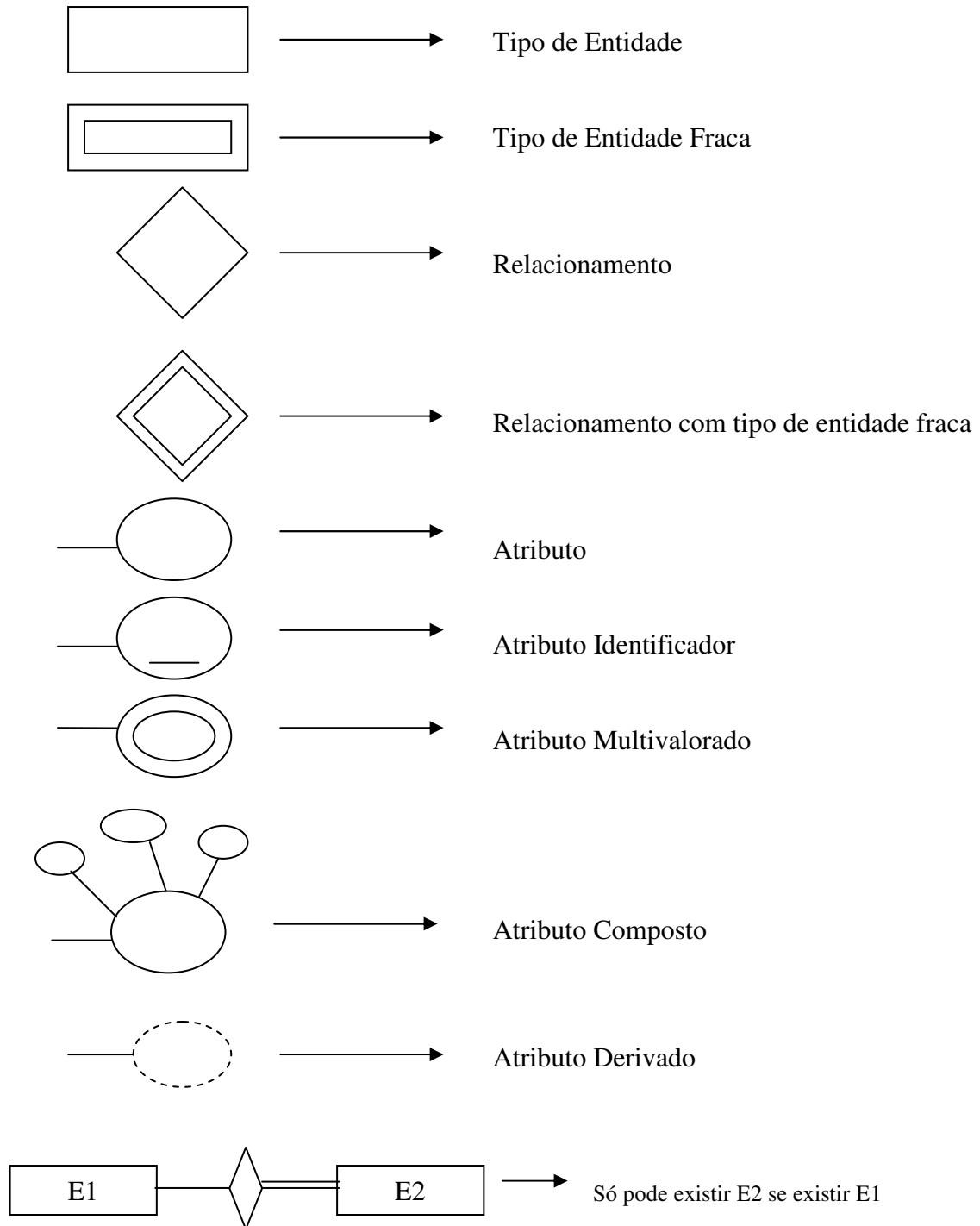


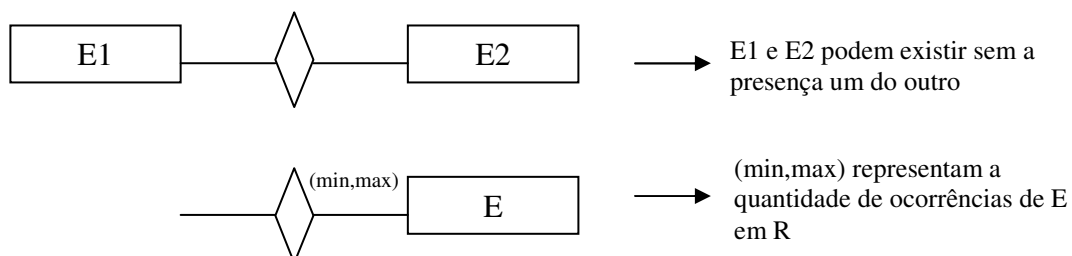
Tanto o relacionamento entre ALUNO e APROVEITAMENTO como o relacionamento entre DISCIPLINA e APROVEITAMENTO são do tipo associativo pois ambos relacionam entidades primárias de ALUNO e DISCIPLINA à entidade do tipo associativa APROVEITAMENTO.

**Auto-Relacionamentos****1.5. Dicionário de Dados**

O dicionário de dados contém as definições das entidades, dos relacionamentos e dos atributos de um modelo de dados.

### 1.6. Convenções para Construção de um “DER”

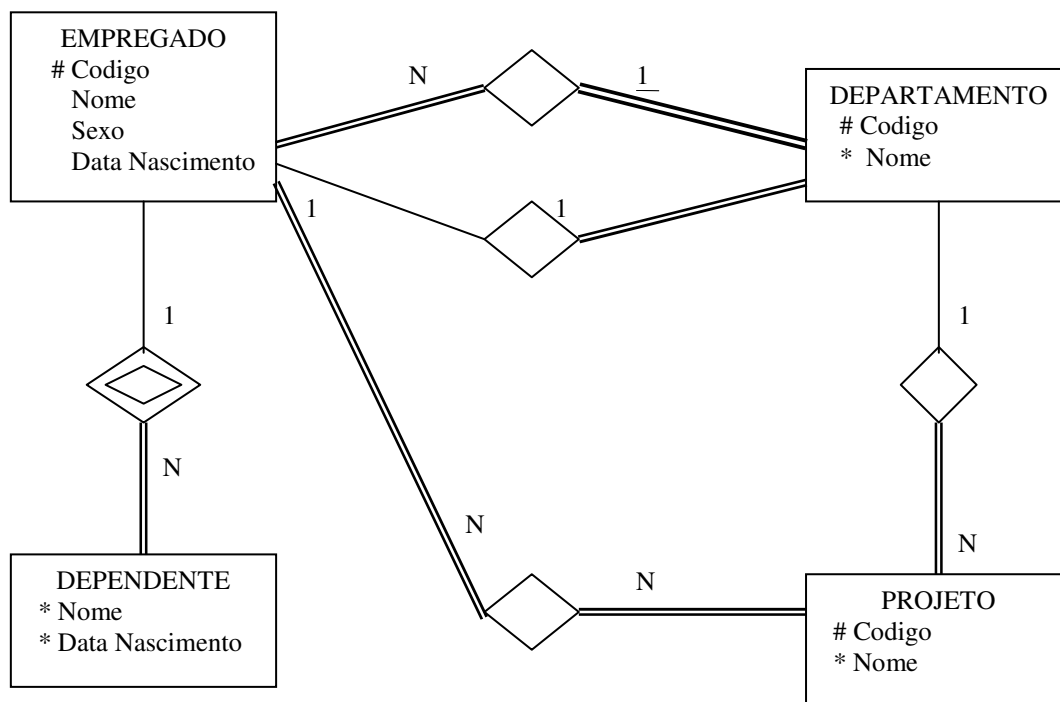




### 1.7. Nomenclatura dos elementos de um DER

Nome do Tipo de Entidade -----> Letras maiúsculas e no singular  
 Nome de Relacionamento -----> Letras maiúsculas e no singular  
 Nome de Atributo -----> Inicia com letra maiúscula  
 Regra -----> Letras minúsculas

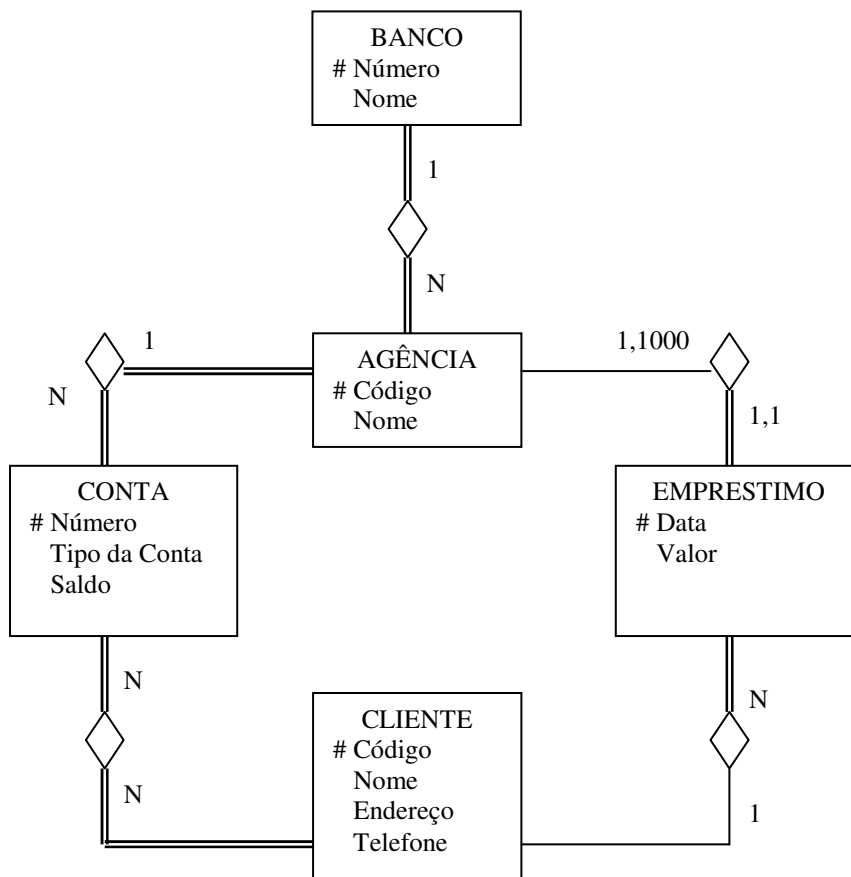
### 1.8. Exemplo de “DER”



## 1.9. Exercícios

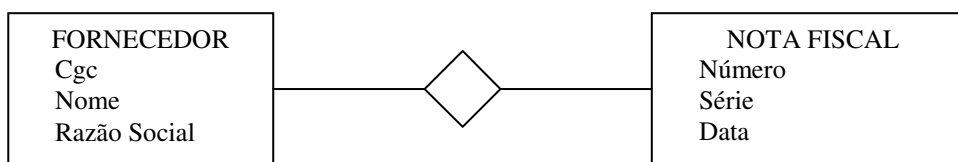
### Exercício 1

Considere parte de um ambiente de banco. Um banco tem várias agências. As contas de uma agência podem se referir a pessoas físicas ou jurídicas. Há contas que podem ter um único titular e contas que podem ter mais de um. Não há contas sem cliente e nenhum cliente da agência sem conta. Uma agência exige necessariamente um banco e não há banco sem agência. As agências fazem empréstimos aos seus clientes há cliente que tem mais de um empréstimo, mas o número de empréstimos de cada agência é limitado a 1000.



### Exercício 2

Quais os possíveis atributos que podem identificar as duas entidades no relacionamento a seguir.



**Exercício 3**

Construir um possível DER a partir do conteúdo do relatório abaixo.

Almoxarifado	Endereço	Código Produto	Nome Produto	Quantidade
234	Rua 15, Centro	A12	Arroz	20
234	Rua 15, Centro	F15	Feijão	150
456	Rua 9, S. Oeste	J14	Açúcar	180
456	Rua 9, S. Oest	F15	Feijão	200

**Exercício 4**

Construir possíveis relacionamentos entre as entidades abaixo e identificar pelo menos dois atributos para cada uma delas. As entidades referem-se a um ambiente hospitalar.

Entidades: PACIENTE, CIRURGIÃO, CIRURGIA, TIPO DE CIRURGIA.

**Exercício 5**

Construir o DER correspondente às entidades abaixo referentes ao ambiente de recursos humanos de uma empresa. Identifique pelo menos dois atributos para cada entidade e indique o identificador de cada uma. Justifique sua resposta.

**Exercício 6**

Construir o DER correspondente a um ambiente escolar com as seguintes entidades: DEPARTAMENTO, ALUNO, DISCIPLINA, TURMA, APROVEITAMENTO, PROFESSOR respondendo as seguintes perguntas:

1. Qual professor leciona cada disciplina?
2. Qual a nota do aluno em determinada disciplina e qual o professor que deu a nota?
3. Em qual turma de qual disciplina o aluno está matriculado e quais os professores dessa turma?
4. Em qual departamento o professor está vinculado?
5. Quais as disciplinas que são de responsabilidade de cada departamento?
6. Qual o horário de aula de cada turma?
7. Quantos são os alunos do sexo masculino e quantos são do sexo feminino?
8. Qual a idade de cada aluno?
9. Quais são os pais de cada aluno?

**Exercício 7**

Que alterações seriam necessárias fazer no DER do exercício anterior, considerando que os pais de um aluno também podem ser estudantes, sem que haja duplicidade de informações no modelo?

**Exercício 8**

Identificar os relacionamentos entre as entidades relacionadas abaixo, e para cada entidade identifique, pelo menos, três atributos. Construa o quadro de definição dos atributos para cada entidade. As entidades se referem ao ambiente de um ponto de comércio varejista (por exemplo, uma loja de calçados).

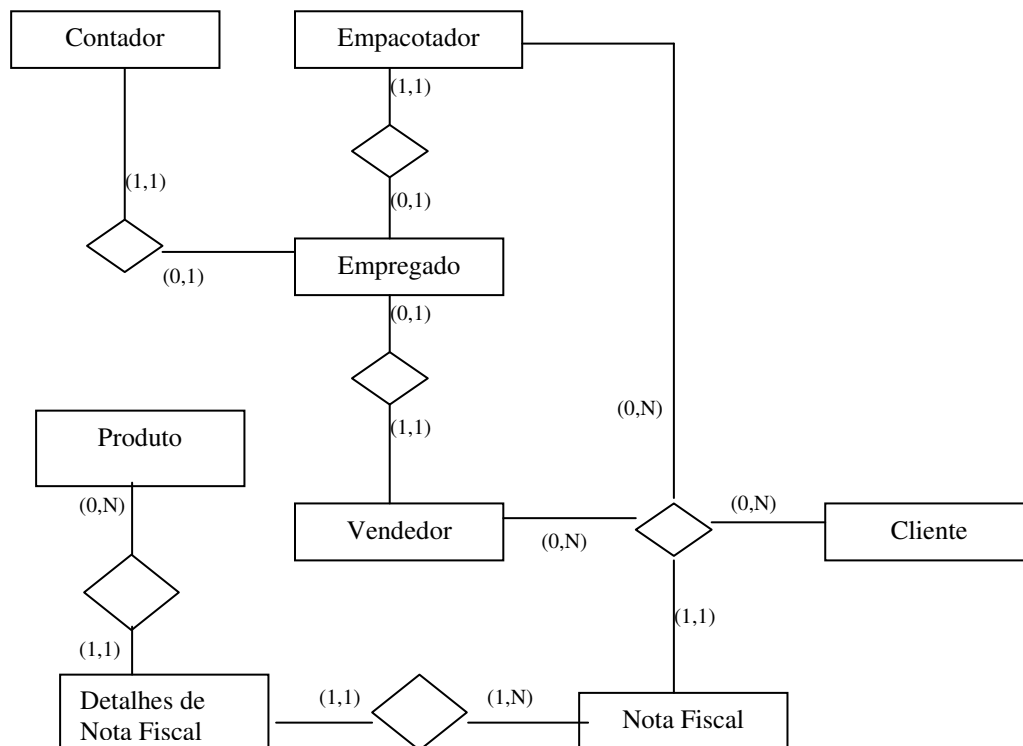
O modelo construído deve responder as seguintes perguntas:

1. Quais Empregados são Empacotadores, Contadores e Vendedores?
2. Qual o Vendedor que atendeu o Cliente?
3. Quem empacotou os produtos comprados para o Cliente?
4. Qual o valor total da compra do Cliente?
5. Quais os Clientes que foram atendidos mas que não compraram nada?
6. Se o Cliente comprou alguma coisa, quais os produtos comprados?
7. Qual o total de vendas de cada Vendedor por dia?
8. Qual o total das vendas correspondentes às mercadorias empacotadas por cada Empacotador?

Relação de entidades:

Empregado, Contador, Empacotador, Vendedor, Cliente, Nota Fiscal, Detalhes de Notas Fiscais (Relação de Produtos) e Produto.

DER proposto





**Definição dos Atributos**

Vendedor

Atributos	ID	OB	MV	AD	AC	CE	NAT	TAM	DEC	DOM
Matrícula	S	S					N	4		> zero
Taxa de Comissão		S					N	4	2	> zero
Telefone			S				C	14		<> Nulo

## 2. Normalização de Dados

Normalização é um processo de depuração de um modelo de dados para reduzir sua redundância e aumentar sua estabilidade.

### CONCEITOS BÁSICOS

#### 1. Dependência Funcional Completa

Quando um atributo não identificador depende do(s) atributo(s) identificador(es).

#### 2. Dependência Funcional Parcial

Quando um atributo não identificador depende de parte dos atributos identificadores.

#### 3. Dependência Funcional Transitiva

Quando um atributo não identificador depende de outro atributo também não identificador.

#### A normalização permite eliminar atributos:

- Com mais de um valor
- Duplicados ou repetidos
- Que contém dados derivados de outros atributos

### PRIMEIRA FORMA NORMAL

- Uma entidade está na primeira forma normal se não tem atributos com mais de um valor, nem atributos que ocorrem mais de uma vez.

### SEGUNDA FORMA NORMAL

- Uma entidade está na segunda forma normal se está na primeira forma normal e todos os seus atributos não identificadores são dependentes do atributo identificador da entidade.

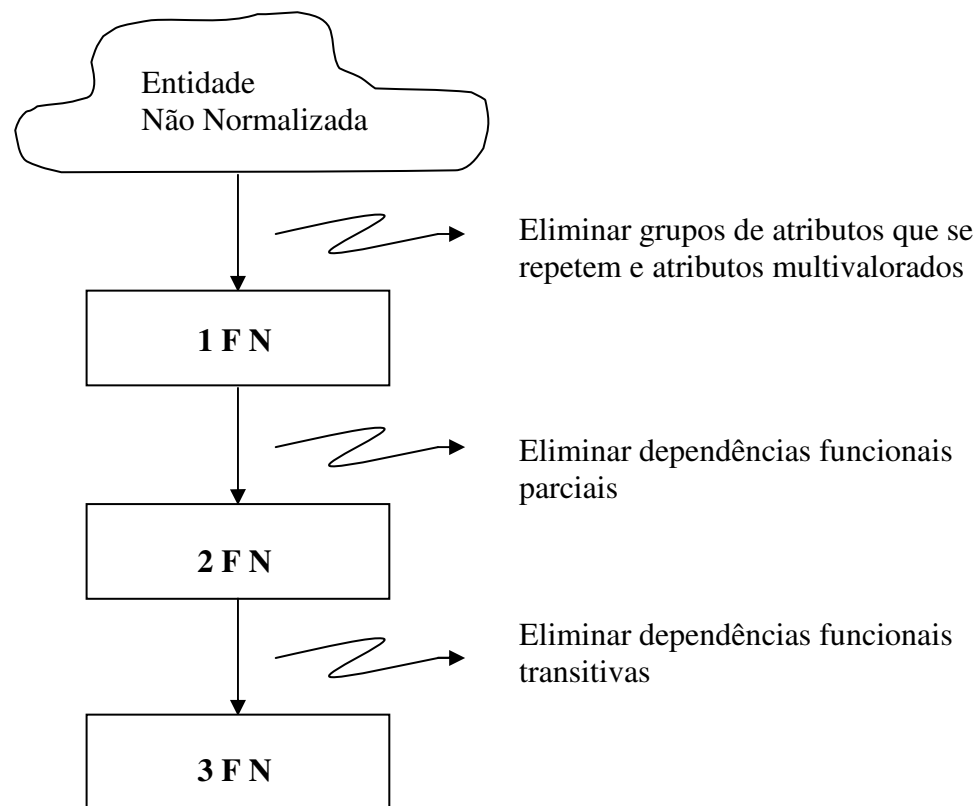
### TERCEIRA FORMA NORMAL

- Uma entidade está na terceira forma normal se está na primeira e na segunda forma normal e não contém atributos não identificadores dependentes de outros atributos não identificadores

#### Observações

- Um modelo de E x R normalizado é convertido facilmente para um Banco de Dados relacional em tempo de projeto.
- A terceira forma normal geralmente é aceita como boa para projeto de Banco de Dados sem redundância.

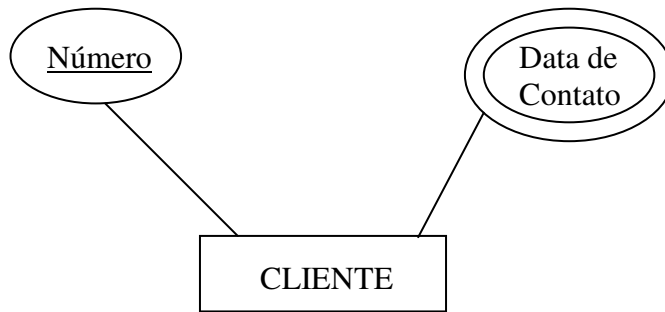
- Existem formas normais de nível maior, mas que geralmente não são usadas.

**PASSOS:****Verificação da Primeira Forma Normal**

- Verificar se cada atributo tem um único valor para cada instância da entidade.
- Nenhum atributo pode ter valores repetidos.

**Exemplo:**

Verificar se a entidade Cliente abaixo está na 1FN. Se não estiver, convertê-la para a 1FN.



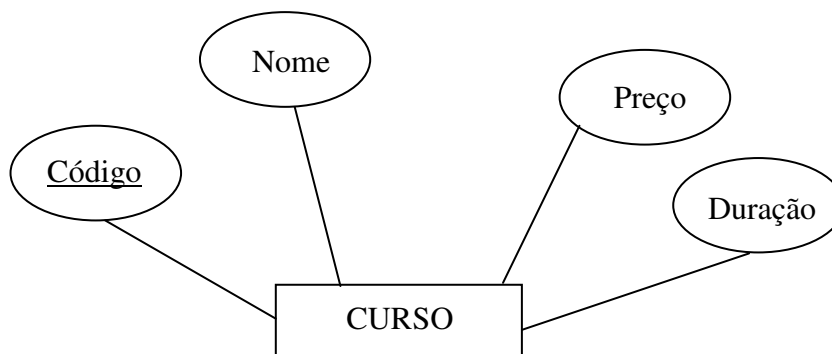
O Atributo *data de contato* pode ter múltiplos valores, portanto a entidade CLIENTE não está na 1FN. Para transformá-la para a 1FN vamos criar uma entidade adicional CONTATO e relacioná-la com um relacionamento 1:M no sentido CLIENTE - CONTATO.

**Verificação da Segunda Forma Normal**

- Verificar se cada atributo é dependente apenas do identificador da entidade.
- Verificar se existe algum atributo dependente apenas de parte do identificador da entidade.

**Exemplo:**

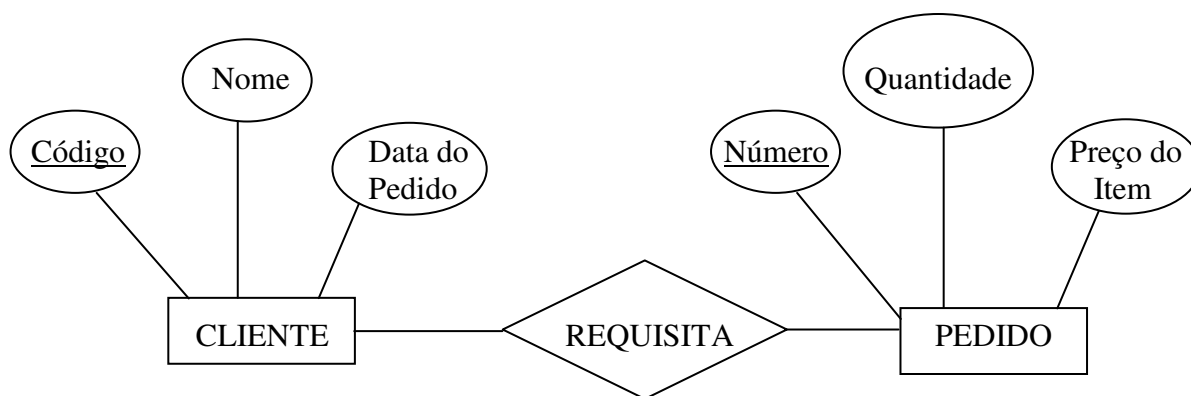
Verificar se entidade CURSO está normalizada.



Cada *código* determina um valor específico para *nome*, *duração* e *preço*, todos eles são dependentes exclusivamente do identificador, e nenhum dos atributos é derivado um do outro. Portanto a entidade está normalizada.

**Exemplo:**

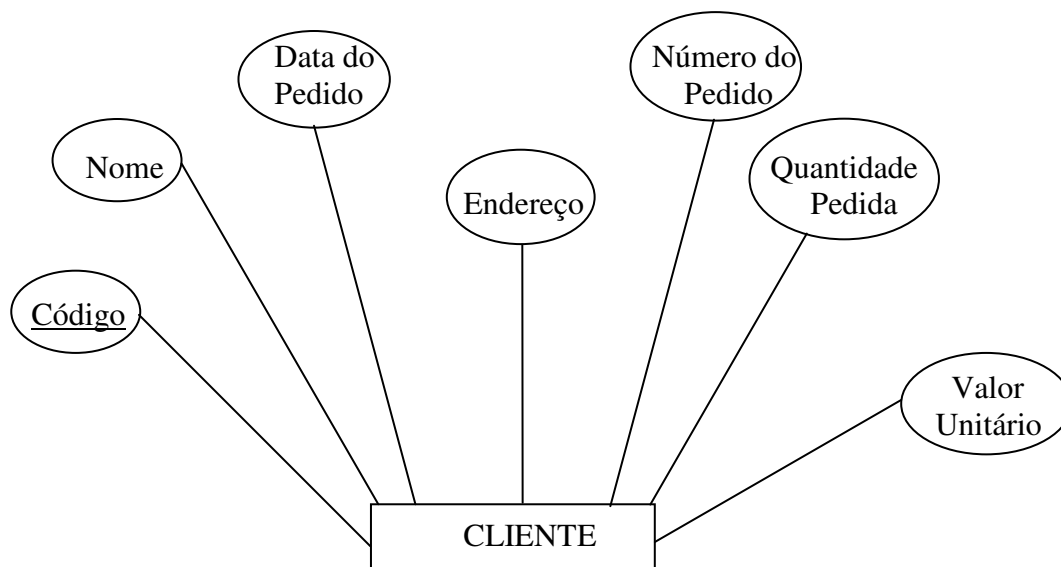
Verificar se as entidades abaixo estão normalizadas.



Cada instância de CLIENTE e PEDIDO determina valores específicos de quantidade e preço do item. O atributo data do pedido está perdido na entidade CLIENTE, porque ele não é dependente do identificador da entidade. Ele deve ser um atributo de PEDIDO.

**Exemplo:**

Normalizar a entidade abaixo:



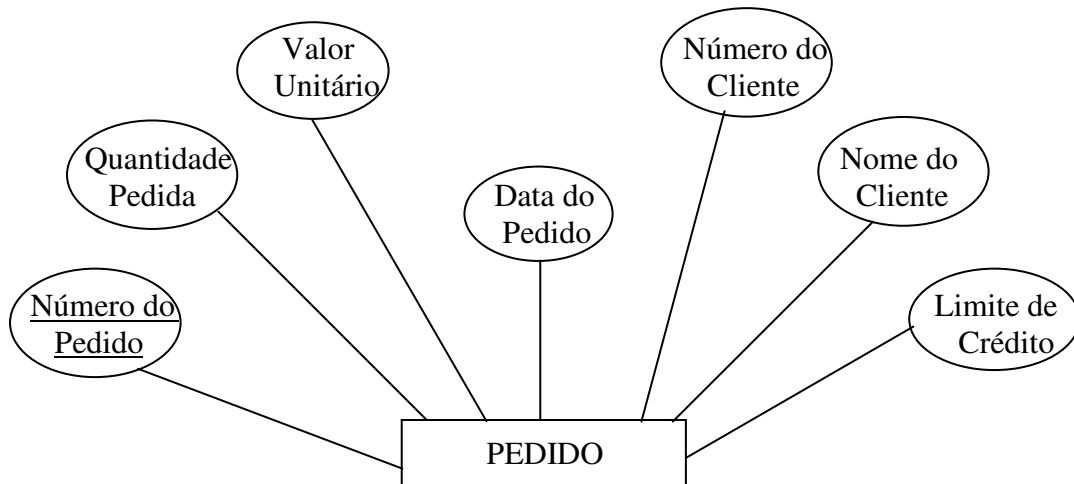
Como a entidade não tem nenhum atributo com valores repetidos ela está na 1FN. Entretanto os atributos data do pedido, número do pedido, quantidade pedida e valor unitário não são dependentes do identificador da entidade, portanto ela não está na 2FN. Para normalizá-la devemos criar uma entidade auxiliar com os atributos não dependentes do identificador.

### **Verificação da Terceira Forma Normal**

- Verificar se existe algum atributo não identificador dependente de outro atributo não identificador.
- Retirar os atributos não identificadores dependentes para uma entidade auxiliar.

#### **Exemplo:**

Verificar se a entidade abaixo está na terceira forma normal.

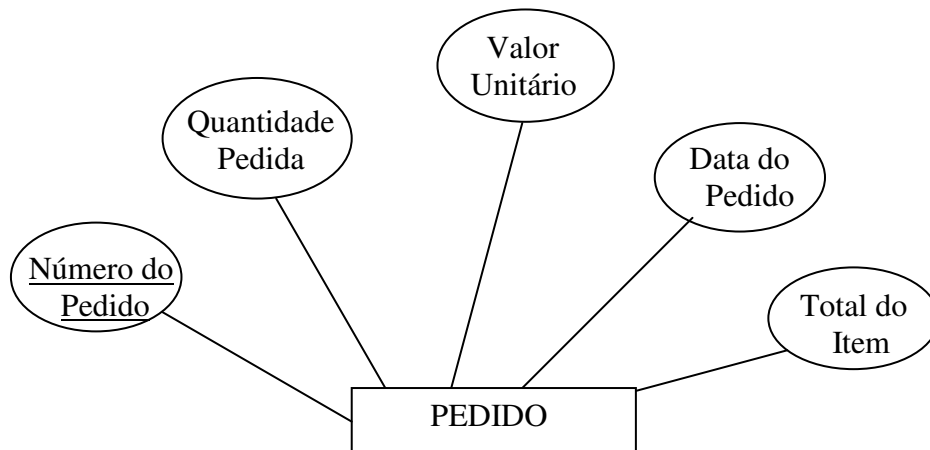


Não existe nenhum atributo com valores repetidos logo a entidade está na 1FN. Os atributos número do cliente, nome do cliente e limite de crédito não são dependentes do identificador da entidade, portanto ela não está na 2FN. Logo a entidade não está na 3FN.

Para passá-la para a 2FN devemos criar uma entidade auxiliar com os atributos não dependentes do identificador.

**Exemplo:**

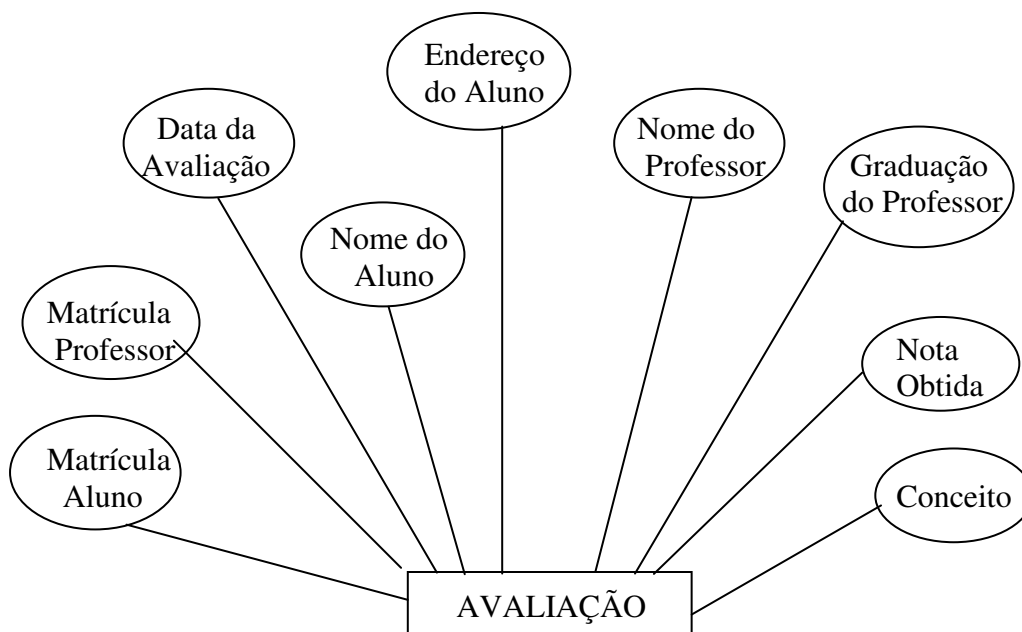
Verificar se a entidade abaixo está na 3FN.



Não existe nenhum atributo com valores repetidos, logo a entidade está na 1FN. Todos os atributos não identificadores são dependentes do identificador da entidade, logo ela está na 2FN. O atributo *total do item* é dependente da *quantidade pedida* e do *valor unitário*, portanto a entidade não está na 3FN. Para passá-la para a 3FN basta eliminar o atributo *total do item* que é desnecessário na entidade.

**EXERCÍCIO:**

Normalizar a entidade abaixo:



### 3. MODELO RELACIONAL

O Modelo Relacional representa a base de dados como uma coleção de relações. Uma relação é concebida como uma tabela de valores, cada linha da tabela representa uma coleção de valores de dados relacionados. Estes valores podem ser interpretados como fatos que descrevem uma entidade ou um relacionamento do mundo real. Os nomes de tabelas e nomes de colunas servem também para auxiliar na interpretação do significado dos valores em cada linha de cada tabela.

Na terminologia do Modelo Relacional, uma linha é chamada TUPLA, o cabeçalho de uma coluna é chamado ATRIBUTO e a tabela é chamada RELAÇÃO. Os tipos de dados que descrevem os tipos de valores que podem aparecer em cada coluna são chamados de DOMÍNIO.

#### DOMÍNIO

Um domínio **D** é um conjunto de valores atômicos, isto é, cada valor do domínio é indivisível segundo o contexto sobre o qual o modelo é concebido. Um método comum de especificar um domínio consiste em especificar o tipo de dado segundo o qual os valores dos dados que formam o domínio são estabelecidos.

#### **Exemplos:**

Alunos\_Matrículas: Conjunto de 5 dígitos válidos como matrícula  
Alunos\_Datas\_Nascimento: Conjunto de datas válidas.

Além de nome, tipo de dado e formato, informações adicionais podem ser fornecidas para facilitar a interpretação dos valores do domínio.

#### **Exemplos:**

Alunos\_Datas\_Nascimento: Conjunto de datas compreendidas entre 01/01/1950 e 31/12/1982.  
Alunos\_Sexo: Deve ser 'M' ou 'F'.

#### RELAÇÃO (Relation Schema R)

Uma relação **R**, denotada por **R(A1, A2, . . . , An)**, representa uma relação de nome **R** e uma lista de atributos **A1, A2, . . . , An**. Cada atributo **Ai** refere-se a um domínio correspondente **D** na relação **R**. **D** é chamado de domínio de **Ai** e é denotado por **dom(Ai)**. O grau da relação **R** é o número de atributos que ela contém.

#### **Exemplo:**

ALUNO ( Matrícula, Nome, DataNascimento, Sexo )

ALUNO é o nome da Relação e seu grau é 4 porque tem quatro atributos. Seus domínios são:

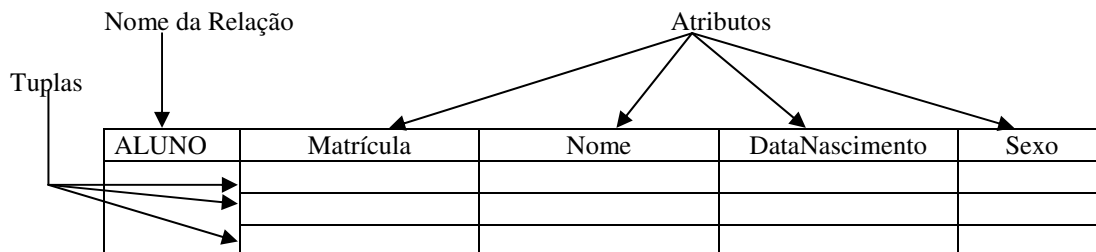
dom ( Matrícula ) = Matrículas  
dom ( Nome ) = Nomes  
dom ( DataNascimento ) = Datas\_Nascimento  
dom ( Sexo ) = Sexos



## TUPLA

Uma instância  $r$  de uma relação  $R(A_1, A_2, \dots, A_n)$ , também denotada por  $r(R)$ , é um conjunto de  $n$ -tuplas  $r = \{ t_1, t_2, \dots, t_m \}$ . Cada  $n$ -tupla  $t$  é uma lista ordenada de  $n$  valores  $t = \langle v_1, v_2, \dots, v_n \rangle$ , onde cada valor  $v_i$ ,  $1 \leq i \leq n$ , é um elemento do domínio  $\text{dom}(A_i)$  ou é nulo.

### Exemplo:



A instância de uma relação em um determinado momento reflete somente as tuplas válidas que representam um estado particular do mundo real. Em geral, como o estado do mundo real muda, muda também a instância da relação, transformando-se em outro estado. Entretanto, o esquema da relação,  $R$ , é relativamente estático e raramente muda, como por exemplo, quando adicionamos um novo atributo para representar uma nova informação que não existia originalmente. É possível que vários atributos, de uma mesma relação ou de relações diferentes, compartilhem o mesmo domínio. Os atributos indicam diferentes interpretações para o domínio. Por exemplo, o domínio *Datas\_Nascimento* para o atributo *DataNascimento* da relação *ALUNO* significa a data de nascimento do aluno, já para a relação *PROFESSOR*, representará a data de nascimento do professor.

## CARACTERÍSTICAS DAS RELAÇÕES

### Ordenação das tuplas na relação

Uma relação é definida como um conjunto de tuplas. Matematicamente, elementos de um conjunto não têm nenhuma ordem entre si, portanto, tuplas em uma relação não têm uma ordem particular. Entretanto, em um arquivo, registros são fisicamente armazenados no disco de tal forma que existe aí uma ordem entre os registros. Esta ordem indica o primeiro, segundo, etc., e o último registro no arquivo. Assim, quando listamos o conteúdo de uma relação como uma tabela, as linhas aparecem em uma determinada ordem.

A ordenação das tuplas não faz parte da definição relacional, porque a relação tenta representar fatos em um nível lógico ou abstrato. Muitas ordens lógicas podem ser especificadas em uma relação.

### Ordenação dos valores na tupla

Analogamente à definição precedente de uma relação, uma  $n$ -tupla é uma lista ordenada de  $n$  valores, assim a ordenação de valores em uma tupla – e portanto de atributos em uma definição de relação – é importante.

Entretanto, em um nível lógico, a ordem dos atributos e seus valores não são realmente importante tanto como a correspondência entre atributos e seus valores.

### Uma definição alternativa de relação

Considerando-se desnecessária a ordenação de valores em uma tupla, pode-se considerá-la como um conjunto de pares (**<atributo>**,**<valor>**), onde cada par fornece o valor para cada atributo.

**Exemplo:**

t = < ( Matrícula, \_\_\_\_\_ ), ( Nome, \_\_\_\_\_ ), ( DataNascimento , \_\_\_\_\_ ), ( Sexo , \_\_ ) >

t = < (Nome, \_\_\_\_\_) , (Sexo, \_\_\_\_\_) , (Matrícula, \_\_\_\_\_) , (DataNascimento, \_\_\_\_\_) >

### Valores nas tuplas

Cada valor em uma tupla é um valor atômico, isto é, indivisível no contexto do modelo relacional. Portanto, atributos compostos e multivalorados **não** são permitidos. Aí está a importância das técnicas de normalização. Os valores de alguns atributos em uma tupla em particular podem ser desconhecidos ou não existirem para esta tupla. Um valor especial, denominado **nulo**, é usado para estes casos.

## Notação para o modelo relacional

Relação R de grau n	→	$R(A_1, A_2, \dots, A_n)$
Tupla t na relação r (R)	→	$t = \langle v_1, v_2, \dots, v_n \rangle$ Onde $v_i$ é o valor correspondente ao atributo $A_i$ .
Nomes de Relações	→	Q, R, S
Instâncias de Relações	→	q, r, s
Tuplas	→	t, u, v

Nomes dos atributos podem ser qualificados com o nome da relação a que pertencem, por exemplo:

ALUNO.Matrícula, ALUNO.Nome, etc.

## RESTRIÇÕES NO MODELO RELACIONAL

### Restrições de domínio

Restrições de domínio especificam que o valor de cada atributo **A** deve ser um valor atômico pertencente ao domínio **dom(A)** para este atributo. Os tipos de dados associados a domínios incluem os numéricos inteiros, reais, caracteres, strings de tamanho fixo, strings de tamanho variável, data, etc. Os domínios podem ainda especificar um conjunto de valores válidos explicitamente identificados.

### Restrições de Chave ( Chave Primária = *Primary Key* )

Uma relação é definida como um conjunto de tuplas. Por definição, todos os elementos do conjunto são distintos, portanto, todas as tuplas em uma relação devem também ser distintas. Isto significa que não pode haver duas tuplas com a mesma combinação de valores para todos os atributos.

### Restrição de Integridade de Entidade

A restrição de integridade de entidade estabelece que o valor para a chave primária não pode ser nulo. Isto porque o valor da chave primária é usado para identificar individualmente tuplas de uma relação.

Restrições de chave e restrições de integridade de entidade são especificadas para cada relação individualmente.

### Restrição de Integridade Referencial

A restrição de integridade referencial é especificada entre duas relações e é usada para manter a consistência entre as tuplas dessas duas relações. A restrição de integridade referencial estabelece que uma tupla em uma relação que se refere a outra relação deve se referir a uma tupla existente da outra relação.

### Chave Estrangeira ( *Foreign Key* )

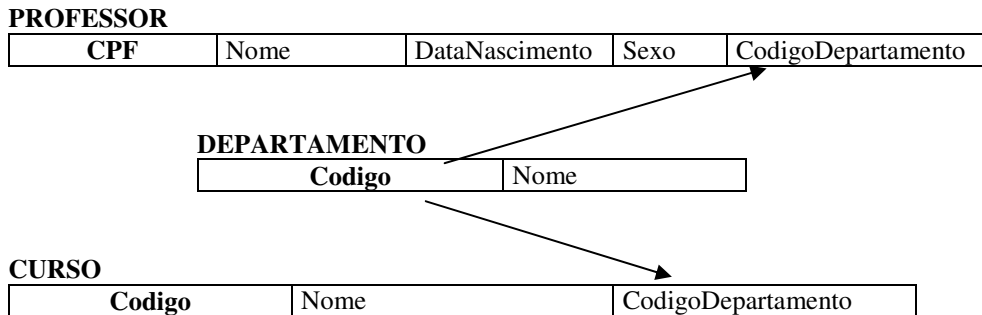
As duas condições citadas a seguir, estabelecem a restrição de integridade referencial entre duas relações R1 e R2. Um atributo, ou conjunto de atributos, FK em uma relação R1, é (são) uma chave estrangeira de R1 se:

1. O(s) atributo(s) FK tem o mesmo domínio do(s) atributo(s) da chave primária PK da relação R2; diz-se que o(s) atributo(s) FK refere(m)-se à relação R2.
2. O(s) valor(es) de FK na tupla t1 de R1 existe(m) como o valor de uma PK em alguma tupla de R2 ou é nulo.

Formalmente: t1 [FK] = t2 [PK].

### Importante

→ Em um banco de dados de muitas relações, existem geralmente muitas restrições de integridade referencial. Para especificar estas restrições, deve-se primeiro ter uma clara compreensão do significado do papel de cada atributo das várias relações do esquema do banco de dados. As restrições de integridade referencial surgem dos relacionamentos entre as entidades.

**Exemplo:**

→ Todas as restrições de integridade devem ser especificadas se desejamos manter as restrições para todas as instâncias do banco de dados. Portanto, em um sistema relacional, a linguagem de definição de dados (DDL) deve prover recursos para especificar os vários tipos de restrições para que o sistema gerenciador do banco de dados possa cumpri-los automaticamente.

#### 4. Algoritmo de Mapeamento ER / RELACIONAL

**Passo 1** – Para cada tipo de entidade forte E no esquema ER, crie uma relação R e inclua todos os atributos simples de E. Inclua somente os atributos componentes simples de cada atributo composto. Escolha uma dos atributos identificadores de E como chave primária de R. Se a chave escolhida for composta, o conjunto de atributos simples que a formam, juntos formarão a chave primária de R.

**Passo 2** – Para cada tipo de entidade fraca F no esquema ER cujo tipo de entidade forte identificadora é E, crie uma relação R, e inclua todos os atributos simples (ou componentes simples de atributos compostos) de F como atributos de R. Além disso, inclua como chave estrangeira em R os atributos que formam a chave(s) primária(s) da(s) relação(ões) que correspondem ao tipo de entidade forte identificadora. Esse procedimento mapeia o relacionamento identificador de F. A chave primária de R é a combinação da(s) chave(s) primária(s) da(s) entidade(s) forte(s) e da chave parcial do tipo de entidade fraca F, se existir.

**Passo 3** – Para cada tipo de relacionamento R binário 1:1 no esquema ER, identifique as relações S e T que correspondam aos tipos de entidades participantes de R. Escolha uma das relações – digamos, S – e inclua como chave estrangeira em S a chave primária de T. É melhor escolher um tipo de entidade com participação total em R para desempenhar o papel de S. Inclua todos os atributos simples (ou componentes simples de atributos compostos) do tipo de relacionamento R 1:1 como atributos de S.

**Passo 4** – Para cada tipo de relacionamento binário R 1:N que não seja identificador de entidade fraca, identifique a relação S que representa o tipo de entidade participante no lado N do tipo de relacionamento. Inclua como chave estrangeira de S a chave primária da relação T que representa o outro tipo de entidade participante de R; isto se deve ao fato de que cada instância de entidade do lado N se relaciona a, no máximo, uma instância de entidade do lado 1 do tipo de relacionamento. Inclua todos os atributos simples (ou componentes simples de atributos compostos) do tipo de relacionamento 1:N como atributos de S.

**Passo 5** – Para cada tipo de relacionamento binário  $R\ M:N$ , crie uma nova relação  $S$  para representar  $R$ . Inclua como chaves estrangeiras em  $S$  os atributos que formam as chaves primárias das relações que representam os tipos de entidades participantes de  $R$ ; a combinação dessas chaves estrangeiras formarão a chave primária de  $S$ . Também inclua todos os atributos simples do tipo de relacionamento  $M:N$  (ou componentes simples de atributos compostos) como atributos de  $S$ .

**Passo 6** – Para cada atributo multivalorado  $A$ , crie uma nova relação  $R$  que inclui um atributo correspondente a  $A$  mais a chave primária  $K$  (como chave estrangeira em  $R$ ) da relação que representa o tipo de entidade ou tipo de relacionamento que tem  $A$  como atributo. A chave primária de  $R$  é a combinação de  $A$  com  $K$ . Se o atributo multivalorado é composto, então inclua somente seus componentes simples.

**Passo 7** – Para cada tipo de relacionamento  $n$ -ário  $R$ , com  $n > 2$ , crie uma nova relação  $S$  para representar  $R$ . Inclua como chaves estrangeiras em  $S$  as chaves primárias das relações que representam os tipos de entidades participantes de  $R$ . Também inclua todos os atributos simples (ou componentes simples de atributos compostos) como atributos de  $S$ . A chave primária de  $S$  é, normalmente, a combinação de todas as chaves estrangeiras que referenciam as relações que representam os tipos de entidades participantes de  $R$ . No entanto, se a restrição de participação (min,max) de um dos tipos de entidades participantes de  $R$  (digamos,  $E$ ) tiver  $\max = 1$ , então a chave primária de  $S$  pode ser simplesmente a chave estrangeira que referencia a relação  $E'$  correspondente a  $E$ , pois, nesse caso, cada entidade  $e$  em  $E$  participará em, no máximo, uma instância de relacionamento de  $R$  e, portanto, pode identificar unicamente essa instância de relacionamento.

## 5. Exercícios de Modelagem de Dados

### a. Construir o DER correspondente à descrição dos relacionamentos abaixo:

- Um Item de Pedido de Compra deve se referir a um único Pedido de Compra.
- Um Pedido de Compra deve se referir a um único Fornecedor.
- Um Item de Pedido de Compra deve se referir a um e somente um Produto.
- Um Pedido de Compra deve conter um ou vários Itens de Pedido de Compra.
- Um Fornecedor pode ter vários Pedidos de Compra a ele solicitado.
- Um Produto pode ser referido por nenhum ou vários Itens de Pedido de Compra.
- Um Produto deve ser fornecido por um ou vários Fornecedores.
- Um Fornecedor deve fornecer um ou vários Produtos.
- Um Pedido de Compra pode ser atendido por uma ou várias Notas Fiscais.
- Uma Nota Fiscal deve se referir a um e somente um Pedido de Compra.
- Uma Nota Fiscal deve ser constituída de um ou mais Itens de Nota Fiscal.
- Um Item de Nota Fiscal deve se referir a uma única Nota Fiscal.

### b. Construir o DER e definir todos os seus atributos para o problema a seguir:

Sou gerente de uma empresa de treinamento que ministra vários cursos de caráter técnico. Ministramos vários cursos que são identificados por um código, nome e preço. Os cursos “Introdução ao UNIX” e “Programando em C” são alguns de nossos cursos mais populares. A duração de cada curso pode variar de um a quatro dias. Um instrutor pode ensinar vários cursos. Paul Rogers e Maria Gonzales são dois de nossos melhores instrutores. Mantemos aqui o nome e o telefone de cada instrutor. Nós criamos um curso e alocamos um instrutor. Os alunos (clientes) podem participar de vários cursos e vários deles o fazem. O Jamie Brown, da Docegeo, assiste a todo curso que oferecemos. Além do nome, mantemos também, o número do telefone dos alunos. Alguns de nossos alunos e instrutores não possuem telefone.

### c. Construir um DER e definir todos os seus atributos para o problema a seguir: (Profa. Karin Becker da PUCRS)

Uma firma vende produtos de limpeza. Cada produto é caracterizado por um código único, nome do produto, categoria (e.g. detergente, sabão em pó, sabonete, etc.), e seu preço. A categoria é uma classificação criada pela própria firma. A firma possui informações sobre todos os seus clientes. Cada cliente é identificado por um código único (também interno à firma), o nome do cliente, endereço (rua, número, sala, cidade, CEP, UF), telefone, o *status* do cliente (“bom”, “médio”, “ruim”), e o seu limite de crédito. Guarda-se igualmente a informação dos pedidos feitos pelos clientes. Cada pedido possui um número e guarda-se a data de elaboração do pedido. Cada pedido pode envolver de um a vários produtos e para cada produto, indica-se a quantidade pedida.

**d. Construir um DER e definir todos os seus atributos para o problema a seguir:**

Sou proprietário de uma pequena loja de vídeo. Temos mais de 3.000 fitas aqui e queremos um sistema para controlá-las.

Cada fita contém um número. Para cada filme, precisamos saber seu título e categoria (comédia, suspense, terror, etc.). Muitos de nossos filmes têm mais de uma cópia. A cada cópia, fornecemos um identificador (ID) e então controlamos qual filme uma fita contém. O formato de uma fita pode ser BETA ou VHS. Sempre temos uma fita para cada filme e cada fita tem apenas um filme. Não temos aqui nenhum filme que requeira mais de uma fita.

Freqüentemente as pessoas alugam filmes pelos atores. John Wayne e Katheerine Hepburn são muito populares.

Queremos manter informações sobre os astros que atuam em nossos filmes. Nem todos os filmes são estrelados por astros e só mantemos aqui astros que atuam em filmes do nosso catálogo. Os clientes gostam de saber a data de nascimento de um astro, bem como seu verdadeiro nome.

Temos muitos clientes. Apenas alugamos filmes para pessoas inscritas em nosso Vídeo Clube. Para cada membro do clube mantemos seu primeiro e último nome, telefone e endereço. Claro que cada membro possui um número de título. Além disso, mantemos o *status* de crédito de cada um.

Queremos controlar os aluguéis de filmes. Um cliente pode alugar vários filmes ao mesmo tempo. Apenas mantemos os aluguéis correntes (pendentes). Não controlamos histórico de locações.

**e. Construir um DER e definir todos os seus atributos para o problema a seguir :****Um ambiente de controle de veículos**

Uma empresa de grande porte em Goiás, a EMPRESA MODELO S/A, quer fazer o controle de sua frota de veículos. A frota é constituída de vários tipos de caminhões, caminhonetes e carros pequenos, como Parati, Ômega entre outros. De cada veículo, a empresa precisa saber qual o ano de fabricação e modelo, qual o fabricante (se Chevrolet, Volkswagen, Fiat, etc.), o tipo, data da compra, tipo de combustível, quilometragem atual, cor e quantidade de passageiros que o veículo pode transportar. Um veículo pode ser usado por qualquer empregado da empresa inclusive dirigindo o veículo. Entretanto, a empresa tem em seu quadro de funcionários, motoristas que são chamados quando um empregado precisa usar o veículo, mas não sabe dirigir. Em qualquer caso, o responsável pelo veículo durante seu uso é o motorista do veículo. O controle do uso de um veículo é feito através do preenchimento do formulário chamado OCORRÊNCIA DE USO DE VEÍCULO, no qual devem ser registrados os seguintes dados: quilometragem no momento do recebimento do carro para uso, quilometragem no final do uso, o consumo de combustível e o resultado do acerto da viagem. Cada abastecimento deve ser acompanhado de nota fiscal, emitida pelo posto de combustível, demonstrando a quantidade e o valor do abastecimento. Se o uso do veículo for para uma viagem interurbana, o motorista recebe um adiantamento para fazer face aos abastecimentos necessários. Esse adiantamento é feito em função do consumo por quilômetro rodado do veículo. Ao final da viagem, o motorista devolve a diferença dos gastos ou recebe a diferença se, por acaso, as despesas forem maiores que o adiantamento recebido.

Outra questão importante para a empresa, é o controle de manutenção de cada veículo. Essas manutenções são feitas em auto-mecânicas previamente habilitadas. Um veículo é enviado para manutenção a cada 5.000 Km rodados ou de seis em seis meses, o que ocorrer primeiro. Então da manutenção precisa-se saber: o tipo de manutenção (Preventiva, Gratuita, Corretiva), a data da última manutenção e a quilometragem na época da manutenção. Para cada manutenção efetuada, a empresa registra a quantidade de horas e o valor da mão de obra cobrado por tipo, horas de eletricitista, de pintura, de mecânico, etc. Além disso são registrados a quantidade e o tipo de peças usadas para a manutenção (embreagem, pastilhas de freio, farol, etc.). Só interessa os dados da última manutenção efetuada.

**f. Construir um DER e definir todos os seus atributos para o problema a seguir :**

**Um ambiente de treinamento**

Uma empresa que dá treinamento na área de informática tem a seguinte programação semestral de cursos: DBA para o Banco de Dados Oracle, Visual Basic Básico, Visual Basic Avançado, Delphi Básico e outros. Um curso pode ser ministrado várias vezes no semestre. O curso de DBA Oracle para o primeiro semestre de 1999 teve a seguinte programação: a primeira turma de 15/03/99 a 30/03/99 e a segunda turma de 20/06/99 a 05/07/99. Já o curso de Visual Basic Básico foi programado para: a primeira turma no mesmo período da primeira turma de DBA e a segunda turma para o período de 10/08/99 a 30/08/99. O número de alunos matriculados em cada turma é no máximo 30 em razão do número reduzido de computadores para as aulas práticas. Para fazer a matrícula, o aluno precisa apresentar: CPF, nome completo, endereço, telefones de contato e o comprovante do pagamento da taxa de matrícula (que pode ser feito na rede bancária do estado), o curso que pretende fazer e em qual período. No momento da matrícula, a cada aluno é atribuído um número que o identifica entre todos os alunos da turma. Se o aluno, eventualmente, volta a cursar mais cursos, o número dele continua o mesmo. Há que manter o registro para cada aula, de um resumo sucinto da matéria lecionada, da data e da frequência de cada aluno. A empresa tem o interesse em saber o dia da matrícula de cada aluno e em qual banco e agência o aluno fez o pagamento da matrícula. Para cada turma de cada curso é alocado um professor cujos dados de interesse são: CPF, nome, endereço, preço cobrado para ministrar o curso, telefones para contato e quais os cursos que o professor pode ministrar. Outros dados importantes para a empresa são: valor total das matrículas por turma, por curso, por semestre e por ano. Considerar a hipótese do professor ser aluno de cursos que não seja de sua especialidade.

**g. Construir um DER e definir todos os seus atributos o problema a seguir:**

**O problema do transporte interurbano de pessoas**

O sistema de transporte interurbano de pessoas, se baseia no conceito de linhas de ônibus. Uma linha de ônibus é definida pelo estado através de seus órgãos competentes e deve satisfazer as necessidades de deslocamentos de pessoas em uma determinada região. Uma linha deve atender a um certo número de cidades e é explorada por uma única empresa de ônibus, o que é conseguido através de licitação pública. Podem existir linhas diferentes atendendo a um mesmo conjunto de cidades, por exemplo: a linha de ônibus N1 atende as cidades de Goiânia, Nerópolis, Petrolina, Jaraguá, Rialma, Ceres, Uruaçu e Porangatu e é explorada pelo Rápido Araguaia (linha fictícia). Já a linha N2 atende as mesmas cidades, só que em horários diferentes e é explorada pelo Expresso São Luiz (linha fictícia).

Uma empresa de ônibus, por exigência do poder público, é obrigada a escalar dois motoristas por viagem a cada trecho de 600 Km. Um desses motoristas é responsável pela viagem no trecho que é escalado e o outro é o seu



auxiliar. Por exemplo: nas linhas que servem o percurso de Goiânia a São Paulo, as empresas devem escalar 4 motoristas para a viagem, já que a distância a ser percorrida é maior que 600 Km. Dois motoristas conduzem o ônibus até Uberaba e os outros dois, de Uberaba a São Paulo. Outro fato importante é que cada linha tem os pontos certos de parada obrigatória. Caldas Novas, Uberlândia, Uberaba, Ribeirão Preto e Pirassununga são paradas obrigatórias de uma linha Goiânia – São Paulo. Às vezes numa mesma viagem pode ocorrer troca de ônibus. Essa troca não coincide necessariamente com a troca de motoristas, ou seja, pode não ocorrer na mesma cidade. Uma viagem é definida como o percurso do trajeto que define uma linha de ônibus.

Uma passagem, relativa a uma viagem, deve indicar o nome do passageiro, seu número de identidade, a origem e o destino da passagem, data e hora de embarque, o valor e a plataforma de embarque. A origem e o destino das passagens devem, obrigatoriamente, fazer parte das cidades que compõe o trajeto da linha, embora um passageiro possa descer em qualquer ponto do percurso. Não é permitido, a não ser em caso de incidentes que impeçam o tráfego no trajeto da linha, desvio da rota definida. O número de passagens vendidas não pode exceder o número de cadeiras do ônibus, ou seja, não há possibilidade de um passageiro viajar em pé. Atraso na chegada em cada cidade, de mais de uma hora, deve ser comunicado à sede da empresa que responde pela linha. Outra coisa importante, tendo em vista a segurança dos passageiros, é que um motorista só pode ser escalado para uma viagem a intervalos de 72 horas. Uma transgressão a esta norma, que é determinada pelo poder público, pode acarretar pesadas multas para a empresa infratora.

A respeito das linhas, é bom salientar, que elas são definidas pelo percurso e pelo sentido do percurso, ou seja, a linha de Goiânia - São Paulo tem um número diferente da linha São Paulo - Goiânia, embora possam ter o mesmo trajeto e os mesmos pontos de paradas.

## **h. Construir um DER e definir todos os seus atributos para o problema a seguir:**

### **Um ambiente de uma empresa de transporte**

Uma empresa de transporte de cargas faz frete entre vários estados brasileiros. São Paulo, Goiás, Minas Gerais e Mato Grosso são alguns deles. Em cada estado, apenas algumas cidades são atendidas. Por exemplo: em Goiás, apenas as cidades de Goiânia, Rio Verde e Anápolis são atendidas. Uma cidade atendida significa que a empresa transporta mercadorias dessas cidades e para essas cidades. O transporte também se dá entre cidades de estados diferentes.

O valor do frete pode ser pago pelo remetente ou pelo destinatário da mercadoria. Tanto a pessoa que envia a mercadoria como a que recebe, são consideradas clientes da empresa. Eventualmente, empresas também contratam frete. Nesse caso elas são representadas por um representante. O valor do frete pode ser definido em função do peso ou do valor da mercadoria transportada. Como cobrar o frete é uma decisão da empresa transportadora e é definida no momento da contratação do frete. Por exemplo: o transporte de tecidos é cobrado em função do peso e jóias em função do valor. O preço unitário de cada unidade de peso transportada é válido para qualquer cidade atendida. Cobra-se também do cliente o valor do ICMS da mercadoria transportada e o valor do pedágio pago, se existir, o valor do frete peso e frete valor. Uma característica da cobrança de ICMS é que cada estado tem sua política própria. Em Goiás por exemplo, quando a mercadoria é transportada para fora do estado a taxa de ICMS é de 17% sobre o valor do frete peso ou frete valor e de 12% para mercadorias transportadas dentro do próprio estado (índices fictícios).

Para cada frete contratado, é produzido o documento Conhecimento de Transporte Rodoviário de Carga que tem o seguinte conteúdo: Número do conhecimento, que nunca se repete, e para o remetente, se pessoa física, nome, endereço, telefone e CPF. E se empresa, nome, endereço telefones do representante e razão social,

inscrição estadual, CGC, endereço da empresa e telefones de contato. Para o destinatário o conhecimento deve conter as mesmas informações do remetente. O Conhecimento deve conter ainda, o preço da mercadoria transportada, o valor do ICMS a ser recolhido aos cofres do estado, o valor do pedágio, se existir, o frete peso ou o frete valor, conforme o caso. A indicação de quem paga o frete é necessária, se o remetente ou o destinatário do frete e o peso. Precisa-se também da data em que foi realizado o frete, para efeito de registro contábil da empresa transportadora.

E por fim, deve-se saber também, qual foi o funcionário da empresa responsável pela emissão do Conhecimento de Transporte Rodoviário de Carga, neste caso, apenas o número do registro do empregado na empresa e o seu nome.

#### **i. Construir um DER e definir todos os seus atributos para o problema a seguir:**

##### **Um ambiente rural**

A Secretaria de Agricultura do Estado de Goiás quer cadastrar todos os imóveis rurais do estado (fazendas, chácaras, sítios, etc.) para ter informações da produção de alimentos por ano. A produção por ano seria o somatório da produção de todas as propriedades rurais do estado. São necessárias informações a respeito das propriedades tais como: dono ou donos da propriedade (apenas os donos atuais do imóvel), data da aquisição, área em hectares, município onde está situado o imóvel, qual o preço de aquisição, a distância do município onde está situada e se existir empregados trabalhando na fazenda, o nome e a data de nascimento devem ser armazenados. Precisa-se saber quais os produtos que a propriedade produz, período provável de colheita, quantidade a colher prevista e quantidade efetivamente colhida, além do período de colheita efetivo. Há propriedade que produz vários tipos de produtos, como por exemplo, a Fazenda Macambira, de Goiânia, de propriedade do Sr. Francisco Sá Júnior que produz feijão, soja e milho. Já a Fazenda Maricá, de Pires do Rio, que pertence ao Sr. Antônio de Pádua que tem como sócio o Sr. Olinto Fraga, produz arroz, beterraba e cenoura. É interessante salientar que um imóvel rural pode ser propriedade de uma pessoa jurídica, como por exemplo a Fazenda Ribeirão das Águias, município de Formosa, no estado de Goiás, que é propriedade da empresa Produtora de Grãos Ltda., cujos donos são Rodrigo Machado e João Ferreira, neste caso, há que saber quais os donos da empresa. Como há proprietários que são casados é preciso conhecer também qual a esposa de cada proprietário de cada imóvel. Do proprietário, se pessoa física, as seguintes informações devem ser armazenadas: nome, carteira de identidade, CPF, data de nascimento e pelo menos três telefones para contato. Das esposas, quando existirem, guardar o nome, CPF, se existir, data de nascimento, data do casamento e carteira de identidade. Se o proprietário é uma pessoa jurídica, as seguintes informações são necessárias: nome da empresa, razão social, CGC, qual o dono ou donos da empresa, inscrição estadual e telefones de contato.

#### **j. O Transporte Aéreo: Construir um Modelo de Dados para o problema descrito abaixo:**

O serviço de transporte aéreo de pessoas, no Brasil, é executado pelas empresas aéreas sob o regime de concessão do Governo Federal. A VASP, A VARIG e TAM, são algumas das empresas que prestam esse tipo de serviço. O sistema funciona com base no conceito de voo. Um voo representa a rota que uma aeronave deve percorrer (cidades onde faz escala) para prestar os seus serviços. Quando o governo concede um voo para uma empresa, ou seja, permite que faça uma determinada rota, dá-lhe um número que identifica esse voo para a empresa e que único dentro do sistema. Então, podemos ter duas empresas aéreas fazendo a mesma rota, mas com números diferentes de voo, por exemplo: A Varig pode fazer a rota Brasília, Rio de Janeiro, São Paulo, Curitiba, Florianópolis e Porto Alegre, com o voo 140, e a Vasp pode fazer a mesma rota com o voo 150. Então nós dizemos, vou a Porto Alegre pelo voo 140 da Varig, ou digo, vou a Porto Alegre pelo voo 150 da Vasp. Para o sistema estamos dizendo que estamos percorrendo a mesma rota porém em companhias aérea diferentes.

Outro conceito importante é o de escala. Para cada cidade da rota que a aeronave pousa, nós dizemos que o voo faz escala nesta cidade. Então no nosso exemplo do voo 140 da Varig nós dizemos que ele faz escalas em: Brasília, Rio de Janeiro, São Paulo, Curitiba, Florianópolis e Porto Alegre. Para efeito de informação, aos passageiros em aeroportos, o gestor do sistema de transporte aérea (infraero) precisa saber de cada escala de cada voo as seguintes informações: qual é a companhia aérea responsável pelo voo, qual foi a última escala desse voo (o que significa querer saber de qual cidade a aeronave está vindo), qual é a próxima escala do voo (o que significa querer saber para qual cidade o voo está indo), qual o horário previsto de chegada do voo, se não houver atraso, e qual o horário de partida do voo. Outra coisa importante que se precisa saber de cada rota, é qual a primeira cidade da rota, o que é o mesmo que querer saber de onde procede o voo. Então, no nosso exemplo de rota, nós dizemos que, se estamos dando informação de São Paulo: o voo 140 da Varig, procedente de Brasília, com destino a Porto Alegre e com escala em Curitiba e Florianópolis, tem chega prevista para tal horas e tantos minutos. Então precisa-se saber em cada escala qual é primeira e última cidade da rota. Aqui o conceito de rota é circular. Então a cidade de Brasília, na rota descrita, segue a de Porto Alegre, embora a cidade de Brasília seja a primeira e a de Porto Alegre seja a última da rota. Diante do que está descrito podemos dizer então que: em uma cidade pode haver várias escalas sendo cada escala de um voo diferente, embora um mesmo voo possa ter várias escalas no mesmo dia mas em horários diferentes. Isso porque, no caso de rotas curtas, um voo pode percorrer a rota mais de uma vez por dia. Uma escala deve se referir sempre á cidade de onde o voo procede e a uma cidade para qual o voo se destina. Uma escala sempre pertence a um determinado voo. Diante do exposto construir um Diagrama de Entidades x Relacionamentos que nos possibilite prestar as informações necessárias em cada aeroporto onde o voo faz escala. Quais sejam: Companhia Aérea responsável pelo voo, quais os voos que fazem escala em uma determinada cidade, qual foi a escala anterior à escala de referência e qual é a próxima escala. Qual a primeira cidade e última cidade da rota do voo. Qual o horário previsto de chegada e partida do voo em cada escala. O DER, deve ser apresentado completo: entidades com seus respectivos atributos, identificadores, relacionamentos e cardinalidades.

**k. A Família Zacharias: Construir o Modelo de Dados representativo do problema abaixo.**

A família do sr. Galisteu Bambini Zacharias quer registrar informações sobre os membros de sua árvore genealógica. A intenção é saber quem é filho de quem, quem é irmão de quem, quem é tio e sobrinho de quem, quem é primo (filhos de irmãos) de quem e quem é casado com quem. São necessárias, também, informações sobre a escolaridade, onde residem e quais membros já faleceram.

No caso dos casamentos (aqui entende-se por casamento o fato de um casal viver maritalmente sob um mesmo teto) é preciso saber: se o casamento foi realizado no civil ou no religioso, ou se é um casamento não oficial. Um casamento é considerado não oficial se ele não foi realizado, nem no civil e nem no religioso. Atentar para o fato de um mesmo casamento poder ser realizado no civil e no religioso, na mesma data, ou em datas diferentes. Há pessoas da família que, sejam homens ou mulheres, podem se casar mais de uma vez, mas nunca um membro pode ter mais de um casamento ao mesmo tempo. O registro da data do casamento é importante. Há casamentos desfeitos, seja por divórcio, morte ou desquite. Para esse fato precisa-se saber a data da dissolução e também a causa (divórcio, desquite ou morte). Por exemplo, o sr. Ricardo Bambini Zacharias, já falecido, foi casado duas vezes. O primeiro casamento durou de maio/1940 a novembro/1955, quando foi encerrado por divórcio do casal, e o segundo durou de março/1955 a janeiro/1970, quando foi desfeito pela morte da esposa.

Sobre os filhos precisa-se saber quem são seus pais e se são filhos consangüíneos ou adotivos. É que há casais, na família, que por não poderem ter filhos legítimos, adotam crianças e as criam como se seus filhos fossem. Aqui nesta família, como se resto em todas as outras famílias brasileiras, há filhos que não são produtos de nenhum casamento. Neste caso, precisa saber qual membro da família é o pai ou a mãe e quais são os filhos nesta condição.

Além das informações particulares para certa categoria de membros da família (médicos, engenheiros, advogados, professores), precisa-se saber de cada elemento: o nome, o endereço completo, a data de nascimento, pelo menos um telefone e a data de falecimento, para aqueles já falecidos. O endereço dos componentes é uma informação importante. Afinal de contas os elementos precisam se corresponderem por meio postal. Sabe-se que alguns componentes da família residem no exterior, e outros são de destino ignorado.

Informações importantes, e que também devem ser registradas, são as relativas à escolaridade. Qual é a escolaridade de cada membro, em quais escolas estudou e em que período. Por exemplo, o sr. Figueiredo Bambini Zacharias, estudou o primário no Grupo Escolar Maria Freitas, de 1960 a 1967, o curso secundário, no Colégio Santa Mônica, de 1968 a 1975, e se formou em Engenharia Civil, pela Universidade Federal de Minas Gerais, em 1980, sendo que o seu ingresso no curso superior se deu em 1976, e concluiu o curso de Mestrado em Estruturas Metálicas, em 1983, na Universidade Federal do Rio de Janeiro, depois de dois anos que ingressou no curso. Para os membros com curso superior, indicar em que eles são formados e qual a sua especialidade. Pode haver algum membro com mais de uma especialidade. Por exemplo, há pessoas formadas em medicina com especialidade em neurologia e clínica geral, há professores, da família, especialistas em ensino infantil e em história das artes, mas há também aqueles formados que não se especializaram. O sr. Francisco Bambini Zacharias, formou em Medicina Veterinária, e nunca mais estudou na vida. Há elementos, inclusive que nunca foi à escola, outros terminaram só o primário, outros conseguiram terminar o curso secundário, e uma pequena minoria conseguiu fazer o curso superior. Não é comum, mas há alguns elementos que têm mais de um curso superior.

No caso de morte de um componente da família, precisa-se saber qual a causa da morte, em que cemitério e em que cidade foi sepultado e qual a data de falecimento. Por exemplo, há morte por acidente automobilístico, por acidente de avião, por doença de câncer, etc. Há morte que têm mais de uma causa, neste caso deve-se registrar apenas a causa principal. Há que saber quais pessoas da família já faleceram, quais ficaram viúvas, qual casal perdeu algum filho por falecimento e quais filhos já perderam o pai e/ou a mãe. Quais irmãos já morreram e quais estão vivos. No caso de morte de um casal com filhos menores, estes filhos são criados por outros casais da família, com se fossem seus filhos. Neste caso, o filho, é um filho adotivo especial. É uma adoção por parentesco.

É bom saber, também, a bebida preferida de cada membro da família, informação importantíssima para preparação de festas familiares, e qual o clube de futebol preferido de cada membro. Há membros que não bebem e não gostam de futebol.

E por fim, registrar a preferência religiosa de cada membro. Lembrando sempre que nem todas as pessoas professam alguma religião, e que ao longo da vida uma mesma pessoa pode optar por religiões diferentes. Em vista disso, é bom saber em qual período uma pessoa professa uma determinada religião. Por exemplo, o sr. Roberto Bambini Zacharias, dos 7 aos 30 anos era Católico Apostólico Romano, dos 31 aos 40 anos não professou nenhuma religião e nos anos restantes de vida ele era Espírita. Nenhuma outra informação tem importância relevante para a família.



# SQL



## 1. CREATE TABLE

Use a instrução CREATE TABLE para definir uma nova tabela e seus campos e restrições de campo

### 1.1. Sintaxe

```
CREATE TABLE tabela (campo1 tipo [(tamanho)] [NOT NULL] [índice1] [, campo2 tipo [(tamanho)] [NOT NULL] [índice2] [, ...]] [, CONSTRAINT índicedemulticampos [, ...]])
```

### 1.2. A instrução CREATE TABLE tem estas partes:

Parte tabela	Descrição O nome da tabela a ser criada.
campo1, campo2	O nome do campo ou campos a serem criados na nova tabela. Uma tabela deve ter pelo menos um campo.
tipo	O tipo de dados de campo na nova tabela.
tamanho	O tamanho do campo em caracteres (somente os campos Texto e Binário).
índice1, índice2	Uma cláusula CONSTRAINT que define um índice de campo único. Para maiores informações vide a cláusula CONSTRAINT deste manual.
Índicedemulticampos	Uma cláusula CONSTRAINT que define um índice de campos múltiplos. Para maiores informações consulte o tópico CONSTRAINT.

### 1.2. Exemplos

- a) Criar a tabela: DEPARTAMENTO

```
CREATE TABLE DEPARTAMENTOS
  (Codigo Text(3) NOT NULL,
   Nome Text(30) NOT NULL,
   CONSTRAINT PrkDep PRIMARY KEY (Codigo))
```

- b) Criar a tabela: CURSOS

```
CREATE TABLE CURSOS
  (Codigo Text(3) NOT NULL,
   Nome Text(30) NOT NULL,
   CodigoDepartamento Text(3) NOT NULL,
   CONSTRAINT PrkCurso PRIMARY KEY (Codigo),
   CONSTRAINT FrkDepCurso FOREIGN KEY (CodigoDepartamento)
   REFERENCES DEPARTAMENTOS (Codigo))
```

- c) Cria a tabela: DISCIPLINAS



```
CREATE TABLE DISCIPLINAS
(CodigoCurso Text(3) NOT NULL,
Numero INTEGER NOT NULL,
Nome Text(30) NOT NULL,
Creditos Byte NOT NULL,
Laboratorio Byte NOT NULL,
Prelecao Byte NOT NULL,
CodigoDepartamento Text(3) NOT NULL,
CONSTRAINT PrkDisciplina PRIMARY KEY (CodigoCurso, Numero),
CONSTRAINT FrkCursoDisciplina FOREIGN KEY (CodigoCurso)
REFERENCES CURSOS (Codigo))
```

d) Criar a Tabela: SEMESTRES

```
CREATE TABLE SEMESTRES
(Ano Integer NOT NULL,
Numero Byte NOT NULL,
CONSTRAINT PrkSemestre PRIMARY KEY (Ano,Numero))
```

e) Criar a Tabela: TURMAS

```
CREATE TABLE TURMAS
(CodigoCurso Text(3) NOT NULL,
NumeroDisciplina INTEGER NOT NULL,
Numero Text(3) NOT NULL,
NumeroSubturma Text (1) NOT NULL,
AnoSemestre Integer NOT NULL,
NumeroSemestre Byte NOT NULL,
CONSTRAINT PrkTurma
PRIMARY KEY (CodigoCurso, NumeroDisciplina, Numero, NumeroSubTurma,
AnoSemestre,NumeroSemestre),
CONSTRAINT FrkDisciplinaTurma
FOREIGN KEY (CodigoCurso,NumeroDisciplina)
REFERENCES DISCIPLINAS (CodigoCurso,Numero),
CONSTRAINT FrkDisciplinaSemestre
FOREIGN KEY (AnoSemestre,NumeroSemestre)
REFERENCES SEMESTRES (Ano,Numero))
```

### 1.3. Comentários

Uma cláusula CONSTRAINT estabelece várias restrições em um campo e pode ser utilizada para estabelecer a chave primária.

### 1.4. ALTER TABLE

Modifica a estrutura de uma tabela depois de ter sido criada com a instrução CREATE TABLE.

### 1.5. Sintaxe

ALTER TABLE tabela {ADD {COLUMN campo tipo[(tamanho)] [NOT NULL] [CONSTRAINT índice] |  
CONSTRAINT índicedemulticampos} | DROP {COLUMN campo | CONSTRAINT nomedoíndice} }

### 1.6. A instrução ALTER TABLE tem estas partes:

Parte	Descrição
Tabela	O nome da tabela a ser alterada.
Campo	O nome do campo a ser adicionado ou excluído da tabela.
Tipo	O tipo de dados de campo.
Tamanho	O tamanho do campo em caracteres (somente os campos Texto e Binário).
Índice	O índice para campo. Consulte o tópico da cláusula CONSTRAINT para maiores informações sobre como construir este índice.
Índicedemulticampos	A definição de um índice de campos múltiplos a ser adicionado à tabela. Consulte o tópico da cláusula CONSTRAINT para maiores informações sobre como construir esta cláusula.
Nomedoíndice	O nome do índice de campo múltiplo a ser removido.

### 2.3. Exemplos

- a) Acrescentar na tabela DEPARTAMENTOS, o atributo *DataDeCriacao* de preenchimento opcional e do tipo data.

```
ALTER TABLE DEPARTAMENTOS
ADD COLUMN DataDeCriacao Date
```

- b) Excluir da tabela DEPARTAMENTOS, o atributo *DataDeCriacao*

```
ALTER TABLE DEPARTAMENTOS DROP COLUMN DataDeCriacao
```

### 2.4. Comentários

Através da instrução ALTER TABLE, você pode alterar uma tabela existente de diversas maneiras. Você pode:

Utilizar ADD COLUMN para adicionar um novo campo à tabela. Você especifica o nome do campo, tipo de dados e (para campos Texto e Binário) um tamanho opcional.

Utilizar **ADD CONSTRAINT** para adicionar um índice de campos múltiplos. Para maiores informações sobre índices de campos múltiplos, consulte o tópico da cláusula **CONSTRAINT**.

Utilizar **DROP COLUMN** para excluir um campo. Você especifica somente o nome do campo.

Utilizar **DROP CONSTRAINT** para excluir um índice de campos múltiplos. Você especifica somente o nome do índice após a palavra reservada **CONSTRAINT**.

Você não pode adicionar ou excluir mais de um campo ou índice de cada vez.

Você pode utilizar **NOT NULL** em um campo único ou dentro de uma cláusula **CONSTRAINT** nomeada que se aplica tanto a uma **CONSTRAINT** de campo único ou campos múltiplos. Contudo, você pode aplicar a restrição **NOT NULL** somente uma vez a um campo pois, senão, ocorrerá um erro em tempo de execução.

### 3. DROP TABLE

Exclui uma tabela existente de um banco de dados ou exclui um índice existente de uma tabela.

#### 3.1. Sintaxe

```
DROP {TABLE tabela | INDEX índice ON tabela}
```

#### 3.2. A instrução DROP tem estas partes:

Parte	Descrição
Tabela	O nome da tabela a ser excluída ou a tabela a partir da qual um índice deve ser excluído.
Índice	O nome do índice a ser excluído da tabela.

#### 3.3. Exemplos

- a) Excluir a Tabela: **TABEXEMPLO** criada pelo comando abaixo

Criando a tabela

```
CREATE TABLE TABEXEMPLO (Codigo Text(3), Nome Text(30), DataNascimento Date,  
CONSTRAINT PrkTabExemplo PRIMARY KEY (Codigo))
```

Excluindo a tabela

```
DROP TABLE TABEXEMPLO
```

- b) Excluir o Índice **IdxNome** criado pelos comandos abaixo:

Criando o índice (Vide comando **CREATE INDEX**)

```
CREATE INDEX IdxNome ON DEPARTAMENTOS (Nome ASC)
```

Excluindo o Índice IdxNome

**DROP INDEX *IdxNome* ON DEPARTAMENTOS**

### 3.4. Comentários

Você deve fechar a tabela para poder excluí-la ou remover um índice dela.

Você também pode utilizar ALTER TABLE para excluir um índice de uma tabela.

Você pode utilizar CREATE TABLE para criar uma tabela e CREATE INDEX ou ALTER TABLE para criar um índice. Para modificar uma tabela, use ALTER TABLE.

## 4. CREATE INDEX

Cria um novo índice em uma tabela existente.

### 4.1. Sintaxe

**CREATE [ UNIQUE ] INDEX índice ON tabela (campo [ASC|DESC][, campo [ASC|DESC], ...])**  
**[WITH { PRIMARY | DISALLOW NULL | IGNORE NULL }]**

### 4.2. A instrução CREATE INDEX tem estas partes:

Parte índice	Descrição O nome do índice a ser criado.
tabela	O nome da tabela existente que conterá o índice.
campo	O nome do campo ou campos a serem indexados. Para criar um índice de campo único, liste o nome do campo entre parênteses após o nome da tabela. Para criar um índice de campos múltiplos, liste o nome de cada campo a ser incluído no índice. Para criar índices descendentes, use a palavra reservada DESC; caso contrário, assume-se que os índices são ascendentes.

### 4.3. Exemplos

- a) Criar um Índice único para o campo nome da tabela: **DEPARTAMENTOS**

**CREATE UNIQUE INDEX IdxNome**  
**ON DEPARTAMENTOS (Nome ASC)**

A cláusula UNIQUE indica que o índice criado não pode Ter valores duplicados.

- b) Criar um índice para o campo nome da tabela DEPARTAMENTOS proibindo a entrada de dados nulos.

**CREATE INDEX IdxNome**

```
ON DEPARTAMENTOS (Nome ASC)
WITH DISALLOW NULL
```

A cláusula WITH DISALLOW NULL proíbe a entrada de dados nulos no campo nome.

- c) Criar um índice para o campo nome da tabela DEPARTAMENTOS que não contenha valores nulos

```
CREATE INDEX IdxNome
ON DEPARTAMENTOS (Nome ASC)
WITH IGNORE NULL
```

A cláusula WITH IGNORE NULL proíbe valores nulos no índice.

#### 4.4. Comentários

Para proibir valores duplicados no campo ou campos indexados de diferentes registros, use a palavra reservada UNIQUE.

Na cláusula WITH opcional, você pode impor regras de validação de dados. Você pode:

Proibir entradas Null no campo ou campos indexados dos novos registros, utilizando a opção DISALLOW NULL

Impedir que registros com valores Null no campo ou campos indexados sejam incluídos no índice utilizando a opção IGNORE NULL.

Designar o campo ou campos indexados como a chave primária utilizando a palavra reservada PRIMARY. Isto significa que a chave é exclusiva e, portanto, você pode omitir a palavra reservada UNIQUE.

Você também pode utilizar a instrução ALTER TABLE para adicionar um índice de campo único ou de campos múltiplos a uma tabela e pode utilizar a instrução ALTER TABLE ou a instrução DROP para remover um índice criado com ALTER TABLE ou CREATE INDEX.

## 5. CONSTRAINT

Uma restrição é semelhante a um índice, embora também possa ser utilizada para estabelecer uma relação com uma outra tabela.

Você utiliza a cláusula CONSTRAINT nas instruções ALTER TABLE e CREATE TABLE para criar ou excluir restrições. Há dois tipos de cláusulas CONSTRAINT: um para criar uma restrição em um campo único e outro para criar uma restrição em mais de um campo.

#### 5.4. Sintaxe

Restrição de campo único:

```
CONSTRAINT nome {PRIMARY KEY | UNIQUE | NOT NULL | REFERENCES tabelaexterna
[(campoexterno1, campoexterno2)]}
```

Restrição de campos múltiplos:

```

CONSTRAINT nome
{PRIMARY KEY (primária1[, primária2 [, ...]]) |
UNIQUE (exclusiva1[, exclusiva2 [, ...]]) |
NOT NULL (nãonulo1[, nãonulo2 [, ...]]) |
FOREIGN KEY (ref1[, ref2 [, ...]])
REFERENCES tabelaexterna [(campoexterno1 [, campoexterno2 [, ...]])}]

```

### 5.5. A cláusula CONSTRAINT tem estas partes:

Parte	Descrição
Nome	O nome da restrição a ser criada.
primária1, primária2	O nome do campo ou campos a ser designado(s) à chave primária.
exclusiva1, exclusiva2 nãonulo1, nãonulo2	O nome do campo ou campos a ser designado(s) como uma chave exclusiva.
ref1, ref2	O nome do campo ou campos que estão restritos a valores não-Null
Tabelaexterna	O nome do campo ou campos de uma chave externa que fazem referência a campos em uma outra tabela.
campoexterno1, campoexterno2	O nome da tabela externa contendo o campo ou campos especificados por campoexterno.
	O nome do campo ou campos na tabelaexterna especificados por ref1, ref2. Você pode omitir esta cláusula se o campo referenciado for a chave primária de tabelaexterna.

### 5.6. Comentários

Você utiliza a sintaxe para uma restrição de campo único na cláusula de definição de campo de uma instrução ALTER TABLE ou CREATE TABLE que se segue imediatamente à especificação do tipo de dados do campo.

Você utiliza a sintaxe para uma restrição de campos múltiplos sempre que utilizar a palavra reservada CONSTRAINT fora de uma cláusula de definição de campo em uma instrução ALTER TABLE ou CREATE TABLE.

Utilizando CONSTRAINT, você pode designar um campo como um dos seguintes tipos de restrições:

Você pode utilizar a palavra reservada UNIQUE para designar um campo como chave exclusiva. Isto significa que não pode haver dois registros em uma tabela que tenham o mesmo valor neste campo. Você pode restringir qualquer campo ou lista de campos como exclusivo. Se uma restrição de campos múltiplos for designada como uma chave exclusiva, os valores combinados de todos os campos no índice devem ser exclusivos, mesmo que dois ou mais registros tenham o mesmo valor em apenas um dos campos.

Você pode utilizar as palavras reservadas **PRIMARY KEY** para designar um campo ou conjunto de campos em uma tabela como uma chave primária. Todos os valores na chave primária devem ser exclusivos e não Nulos e só pode haver uma chave primária para uma tabela.

Você pode utilizar as palavras reservadas **FOREIGN KEY** para designar um campo como uma chave externa. Se a chave primária da tabela externa consistir em mais de um campo, você deverá utilizar uma definição de restrição de campos múltiplos, listando todos os campos referenciais, o nome da tabela externa e os nomes dos campos referenciados na tabela externa, na mesma ordem em que os campos referenciais são listados. Se o campo ou campos referenciados forem a chave primária da tabela externa, você não precisará especificar os campos referenciados — por padrão, o mecanismo de banco de dados se comporta como se a chave primária da tabela externa fosse os campos referenciados.

## 6. INSERT INTO

Adiciona um registro ou múltiplos registros a uma tabela. Isto é chamado de consulta acréscimo.

### 6.4. Sintaxe

Acréscimo de múltiplos registros:

```
INSERT INTO destino [IN bancodedadosexterno] [(campo1[, campo2[, ...]])]
    SELECT [origem.]campo1[, campo2[, ...]]
    FROM expressãodetabela
```

Consulta acréscimo de registro único:

```
INSERT INTO destino [(campo1[, campo2[, ...]])]
    VALUES (valor1[, valor2[, ...]])
```

### 6.5. A instrução INSERT INTO possui as partes a seguir:

Parte	Descrição
destino	O nome da tabela ou consulta à qual acrescentar os registros.
bancodedadosexterno0	O caminho até um banco de dados externo. Para obter uma descrição do caminho, consulte a cláusula IN.
origem	O nome da tabela ou consulta a partir da qual os registros vão ser copiados.
campo1, campo2	Nomes dos campos aos quais os dados serão acrescentados, se se seguirem a um argumento destino, ou os nomes dos campos a partir dos quais os dados serão obtidos, se se seguirem a um argumento origem.
Expressãodetabela	O nome da tabela ou das tabelas das quais os registros são inseridos. Este argumento pode ser um nome de tabela simples ou

valor1, valor2

um composto resultante de uma operação INNER JOIN, LEFT JOIN ou RIGHT JOIN ou uma consulta salva.

Os valores a serem inseridos nos campos específicos do novo registro. Cada valor é inserido no campo que corresponde à posição do valor na lista: valor1 é inserido no campo1 do novo registro, valor2 no campo2, e assim por diante. Você deve separar os valores com uma vírgula e colocar os campos de texto entre aspas (' ').

## 6.6. Comentários

Você pode usar a instrução INSERT INTO para adicionar um único registro a uma tabela usando a sintaxe da consulta acréscimo de registro único como mostrado acima. Nesse caso, o código especifica o nome e o valor de cada campo do registro. Você deve especificar cada um dos campos do registro ao qual será atribuído um valor e um valor para aquele campo. Quando você não especifica os campos, o valor padrão ou Null é inserido para colunas ausentes. Os registros são adicionados ao fim da tabela.

Você pode também usar INSERT INTO para acrescentar um conjunto de registros de uma outra tabela ou consulta utilizando a cláusula SELECT ... FROM, como mostrado acima na sintaxe da consulta acréscimo de múltiplos registros. Nesse caso, a cláusula SELECT especifica os campos a acrescentar à tabela destino especificada.

A tabela origem ou destino pode especificar uma tabela ou uma consulta. Se for especificada uma consulta, o mecanismo de banco de dados Microsoft Jet acrescenta registros a todas as tabelas especificadas pela consulta.

INSERT INTO é opcional, mas, quando incluída, precede a instrução SELECT.

Se a tabela de destino contiver uma chave primária, certifique-se de acrescentar valores exclusivos não-Null ao campo ou campos da chave primária; se não o fizer, o mecanismo de banco de dados Microsoft Jet não acrescentará os registros.

Se você acrescentar os registros a uma tabela com um campo AutoNumeração e quiser renumerar os registros acrescentados, não inclua esse campo. Inclua o campo AutoNumeração na consulta se quiser conservar os valores originais do campo.

Use a cláusula IN para acrescentar os registros a uma tabela em um outro banco de dados.

Para criar uma nova tabela, use a instrução SELECT... INTO em lugar de criar uma consulta criar tabela.

Para descobrir quais registros serão acrescentados antes de executar a consulta acréscimo, primeiro execute e visualize os resultados de uma consulta seleção que use os mesmos critérios de seleção.

Uma consulta acréscimo copia registros de uma ou mais tabelas para outra. As tabelas que contêm os registros que você acrescenta não são afetadas pela consulta acréscimo.

Em vez de acrescentar registros existentes de uma outra tabela, você pode especificar o valor de cada campo em um único registro novo, usando a cláusula VALUES. Se você omitir a lista de campos, a cláusula VALUES deverá incluir valores para todos os campos da tabela; caso contrário, a operação INSERT falhará. Use uma instrução INSERT INTO adicional com uma cláusula VALUES para cada registro adicional que quiser criar.



## 7. DELETE

Remove os registros de uma ou mais tabelas relacionadas na cláusula FROM que satisfaçam à cláusula WHERE.

### 7.4. Sintaxe

```
DELETE [tabela.*] FROM tabela WHERE critérios
```

### 7.5. A instrução DELETE possui as partes a seguir:

Parte	Descrição
tabela.*	O nome opcional da tabela da qual são excluídos registros.
tabela	O nome da tabela da qual são excluídos registros.
Critérios	Uma expressão que determina os registros a ser excluídos.

### 7.6. Comentários

DELETE é especialmente útil quando você quer excluir muitos registros.

Para excluir uma tabela inteira do banco de dados, você pode usar o método Execute com uma instrução DROP. No entanto, se você excluir a tabela a estrutura será perdida. Em contrapartida, quando você usa DELETE, somente os dados são excluídos; a estrutura da tabela e todas as suas propriedades, como atributos de campo e índices, permanecem intactos.

Você pode usar DELETE para remover registros de tabelas que estão em um relacionamento um-para-muitos com outras tabelas. As operações de exclusão em cascata fazem com que os registros em tabelas que estão no lado muitos do relacionamento sejam excluídos quando o registro correspondente no lado um do relacionamento é excluído na consulta. Por exemplo, nos relacionamentos entre as tabelas Clientes e Pedidos, a tabela Clientes está no lado um e a tabela Pedidos está no lado muitos do relacionamento. A exclusão de um registro de Clientes resulta na exclusão dos registros Pedidos correspondentes se a opção de exclusão em cascata estiver especificada.

O comando DELETE exclui registros inteiros, não somente os dados em campos específicos. Se você quiser excluir valores de um campo específico, crie uma consulta atualização que altere os valores para Null.

### 7.7. Importante

Depois de remover os registros utilizando uma consulta exclusão, você não poderá desfazer a operação. Se quiser saber quais registros foram excluídos, examine antes os resultados de uma consulta seleção que use os mesmos critérios e, depois, execute a consulta exclusão.

## 8. UPDATE

Altera valores de campos em uma tabela especificada, com base em critérios especificados.

### 8.4. Sintaxe

UPDATE tabela SET novovalor WHERE critérios;

### 8.5. A instrução UPDATE possui as partes a seguir:

Parte	Descrição
tabela	O nome da tabela contendo os dados que você quer modificar.
Novovalor	Uma expressão que determina o valor a ser inserido em um campo específico dos registros atualizados.
critérios	Uma expressão que determina quais registros serão atualizados. Apenas os registros que satisfazem à expressão são atualizados.

### 8.6. Comentários

UPDATE é especialmente útil quando você quer alterar vários registros ou quando os registros que você quer alterar estão em várias tabelas.

Você pode alterar vários campos ao mesmo tempo. O exemplo a seguir aumenta os valores de Quantia do Pedido em 10% e os valores de Frete em 3% para transportadores no Reino Unido (UK):

```
UPDATE Pedidos
  SET QuantiaDoPedido = QuantiaDoPedido * 1.1, Frete = Frete * 1.03
  WHERE PaísDeDestino = 'UK';
```

#### Importante

UPDATE não gera um conjunto de resultados. Além disso, depois de atualizar os registros usando uma consulta de atualização, você não poderá desfazer a operação. Se quiser saber quais os registros que foram atualizados, examine antes os resultados de uma consulta seleção que usem os mesmos critérios e, depois, execute a consulta de atualização.

Mantenha sempre cópias de backup dos dados. Se você atualizar os registros errados, poderá recuperá-los a partir das cópias.

## 9. SELECT INTO

Cria uma criar tabela.

### 9.4. Sintaxe

```
SELECT campo1[, campo2[, ...]] INTO novatabela [IN bancodedadosexterno] FROM origem
```

### 9.5. A instrução SELECT...INTO possui as partes a seguir:

Parte	Descrição
campo1, campo2	Os nomes dos campos a ser copiados na nova tabela.
novatabela	O nome da tabela a ser criada. Deve ser compatível com às convenções de nomenclatura padrão. Se novatabela for igual ao nome de uma tabela existente, ocorrerá um erro interceptável.
bancodedadosexterno	O caminho para um banco de dados externo. Para obter uma descrição do caminho, consulte a cláusula IN.
origem	O nome da tabela existente a partir da qual os registros são selecionados. Podem ser tabelas simples ou múltiplas ou um consulta.

### 9.6. Comentários

Você pode utilizar consultas criar tabela para arquivar registros, fazer cópias de backup das tabelas ou fazer cópias para exportar para um outro banco de dados, ou para usar como base para relatórios que exibem dados sobre um determinado período de tempo. Por exemplo, você poderia produzir um relatório de Vendas Mensais por Região, executando a mesma consulta criar tabela todos os meses.

Convém definir uma chave primária para a nova tabela. Quando você cria a tabela, os campos na nova tabela herdam o tipo de dados e tamanho de campo de cada campo das tabelas base da consulta, mas nenhuma outra propriedade do campo ou da tabela é transferida.

Para adicionar dados a uma tabela existente, use a instrução INSERT INTO em vez de criar uma consulta acréscimo.

Para descobrir quais registros serão selecionados antes de executar a consulta criar tabela, examine antes os resultados de uma instrução SELECT que use os mesmos critérios de seleção.

## 10. SELECT

Retorna informações do banco de dados como um conjunto de registros.

### 10.4. Sintaxe

```
SELECT [atributo] { * | tabela.* | [tabela.]campo1 [AS alias1] [, [tabela.]campo2 [AS alias2] [, ...]] }
      FROM expressãodetabela [, ...] [IN bancodedadosexterno]
      [WHERE... ]
      [GROUP BY... ]
      [HAVING... ]
      [ORDER BY... ]
      [WITH OWNERACCESS OPTION]
```

### 10.5. A instrução SELECT possui as partes a seguir:

Parte	Descrição
atributo	Um dos atributos a seguir: ALL, DISTINCT, DISTINCTROW ou TOP. Você utiliza o atributo para restringir o número de registros retornados. Se nenhum for especificado, o padrão será ALL.
*	Especifica que todos os campos da tabela ou tabelas especificadas estão selecionados.
tabela	O nome da tabela contendo os campos a partir dos quais os registros são selecionados.
campo1, campo2	Os nomes dos campos contendo os dados que você deseja recuperar. Se você incluir mais de um campo, eles serão recuperados na ordem listada.
alias1, alias2	Os nomes a serem utilizados como cabeçalhos de coluna em lugar dos nomes originais de coluna em tabela.
expressãodetabela	O nome da tabela ou tabelas contendo os dados que você deseja recuperar.
bancodedadosexterno	nome do banco de dados que contém as tabelas em expressãodetabela se elas não estiverem no banco de dados atual.

### 10.6. Comentários

Para executar esta operação, o mecanismo de banco de dados Microsoft Jet procura a tabela ou tabelas especificadas, extrai as colunas escolhidas, seleciona as linhas que atendem aos critérios e classifica ou agrupa as linhas resultantes na ordem especificada.

As instruções SELECT não alteram os dados no banco de dados.

SELECT é geralmente a primeira palavra em uma instrução SQL. A maioria das instruções SQL são instruções SELECT ou SELECT...INTO.

A sintaxe mínima para uma instrução SELECT é:

```
SELECT campos FROM tabela
```

Você pode utilizar um asterisco (\*) para selecionar todos os campos em uma tabela. O exemplo a seguir seleciona todos os campos na tabela Funcionários:

```
SELECT * FROM Funcionários;
```

Se um nome de campo for incluído em mais de uma tabela na cláusula FROM, coloque antes dele o nome da tabela e o operador . (ponto). No exemplo a seguir, o campo Departamento está tanto na tabela Funcionários quanto na tabela Supervisores. A instrução SQL seleciona os departamentos a partir da tabela Funcionários e os nomes de supervisores a partir da tabela Supervisores:

```
SELECT Funcionários.Departamento, Supervisores.SupvNome  
FROM Funcionários INNER JOIN Supervisores  
WHERE Funcionários.Departamento = Supervisores.Departamento;
```

Sempre que utilizar funções agregadas ou consultas que retornam nomes de objetos Field ambíguos ou duplicados, você deve utilizar a cláusula AS para fornecer um nome alternativo para o objeto Field. O exemplo a seguir utiliza o título ContagemDePessoas para nomear o objeto Field retornado no objeto Recordset resultante:

```
SELECT COUNT(CódigoDoFuncionário)  
AS ContagemDePessoas FROM Funcionários;
```

Você pode utilizar as outras cláusulas em uma instrução SELECT para ampliar a restrição e organizar os dados retornados. Para maiores informações, consulte o tópico da Ajuda para a cláusula que você está utilizando.

## 11. Atributos ALL, DISTINCT, DISTINCTROW e TOP

Especifica registros selecionados com consultas SQL.

### 11.4. Sintaxe

```
SELECT [ALL | DISTINCT | DISTINCTROW | [TOP n [PERCENT]]]  
FROM tabela
```

### 11.5. Uma instrução SELECT contendo estes atributos possui as partes a seguir:

Parte	Descrição
ALL	Adotada quando você não inclui um dos atributos. O mecanismo de banco de dados Microsoft Jet seleciona todos os registros que atendam às condições na

instrução SQL. Os dois exemplos a seguir são equivalentes e retornam todos os registros da tabela Funcionários:

```
SELECT ALL *  
FROM Funcionários  
ORDER BY CódigoDoFuncionário;
```

```
SELECT *  
FROM Funcionários  
ORDER BY CódigoDoFuncionário;
```

**DISTINCT** Omite registros que contêm dados duplicados nos campos selecionados. Para serem incluídos nos resultados da consulta, os valores de cada campo listado na instrução SELECT devem ser exclusivos. Por exemplo, vários funcionários listados em uma tabela Funcionários podem ter o mesmo sobrenome. Se dois registros contiverem Smith no campo Sobrenome, a instrução SQL a seguir retornará somente um deles:

```
SELECT DISTINCT Sobrenome  
FROM Funcionários;
```

Se você omitir DISTINCT, esta consulta retornará os dois registros Smith. Se a cláusula SELECT contiver mais de um campo, a combinação de valores de todos os campos deverão ser exclusivos para que um dado registro seja incluído nos resultados. A saída de uma consulta que utiliza DISTINCT não é atualizável e não reflete alterações subsequentes feitas por outros usuários.

**DISTINCTROW** Omite dados baseado em registros duplicados completos, e não somente campos duplicados. Por exemplo, você poderia criar uma consulta que associasse as tabelas Cliente e Pedidos no campo CódigoDoCliente. A tabela Clientes não contém campos CódigoDoCliente duplicados, mas a tabela Pedidos contém, pois cada cliente pode fazer vários pedidos. A instrução SQL a seguir mostra como você pode utilizar DISTINCTROW para produzir uma lista de empresas que têm pelo menos um pedido, mas sem exibir detalhes sobre esses pedidos:

```
SELECT DISTINCTROW NomeDaEmpresa  
FROM Clientes INNER JOIN Pedidos  
ON Clientes.CódigoDoCliente = Pedidos.CódigoDoCliente  
ORDER BY NomeDaEmpresa;
```

Se você omitir DISTINCTROW, esta consulta produzirá várias linhas para cada empresa que tenha mais de um pedido. DISTINCTROW produz um efeito somente quando você seleciona campos de algumas, mas não todas, as tabelas utilizadas na consulta. DISTINCTROW será ignorado se a consulta incluir somente uma tabela ou se você obtiver saída de campos de todas as tabelas.

**TOP n [PERCENT]** Retorna um certo número de registros que caem no topo ou na base de um intervalo especificado por uma cláusula ORDER BY. Suponha que você deseje obter os nomes dos 25 melhores estudantes da classe de 1994:

```
SELECT TOP 25 Nome, Sobrenome
FROM Estudantes
WHERE AnoDeGraduação = 1994
ORDER BY MédiaDeNotas DESC;
```

Se você não incluir a cláusula **ORDER BY**, a consulta retornará um conjunto arbitrário de 25 registros da tabela **Estudantes** que satisfaçam à cláusula **WHERE**. O atributo **TOP** não escolhe entre valores iguais. No exemplo anterior, se a vigésima quinta e a vigésima sexta melhores médias de notas forem iguais, a consulta retornará 26 registros. Você também pode utilizar a palavra reservada **PERCENT** para retornar uma certa porcentagem de registros que se situem no topo ou na base de um intervalo especificado pela cláusula **ORDER BY**. Suponha que, em vez dos 25 melhores estudantes, você queira os 10% inferiores da classe:

```
SELECT TOP 10 PERCENT Nome, Sobrenome
FROM Estudantes
WHERE AnoDeGraduação = 1994
ORDER BY MédiaDeNotas ASC;
```

O atributo **ASC** especifica um retorno de valores inferiores. O valor que se segue a **TOP** deve ser um **Integer** sem sinal. **TOP** não afeta o fato de a consulta ser ou não atualizável.

tabela

O nome da tabela a partir da qual os registros são recuperados.

## 12. Cláusula FROM

Especifica as tabelas ou consultas que contêm os campos listados na instrução **SELECT**.

### 12.1. Sintaxe

```
SELECT listadecampos
FROM expressãodetabela [IN bancodedadosexterno]
```

### 12.2. Uma instrução SELECT contendo uma cláusula FROM possui as partes a seguir:

Parte	Descrição
listadecampos	O nome do campo ou campos a serem recuperados juntamente com quaisquer aliases de nome de campo, funções agregadas SQL, atributos de seleção ( <b>ALL</b> , <b>DISTINCT</b> , <b>DISTINCTROW</b> ou <b>TOP</b> ) ou outras opções da instrução <b>SELECT</b> .
expressãodetabela	Uma expressão que identifica uma ou mais tabelas a partir das quais os dados são recuperados. A expressão pode ser um simples nome de tabela, um nome de consulta salvo ou uma composição resultante de um <b>INNER JOIN</b> , <b>LEFT JOIN</b> , ou <b>RIGHT JOIN</b> .

`bancodedadosexterno` O caminho completo de um banco de dados externo contendo todas as tabelas em expressão `detabela`.

### 12.3. Comentários

FROM é exigido e segue-se a qualquer instrução SELECT.

A ordem dos nomes de tabela em expressão `detabela` não é importante.

Para um melhor desempenho e facilidade de utilização, convém utilizar uma tabela vinculada em vez de uma cláusula IN para recuperar dados de um banco de dados externo.

O exemplo a seguir mostra como você pode recuperar os dados da tabela Funcionários:

```
SELECT Sobrenome, Nome
FROM Funcionários;
```

## 13. Cláusula GROUP BY

Combina registros com valores idênticos na lista de campos especificada em um único registro. Um valor de resumo é criado para cada registro se você incluir uma função agregada SQL, como Sum ou Count, na instrução SELECT.

### 13.1. Sintaxe

```
SELECT listadecampos
      FROM tabela
      WHERE critérios
      [GROUP BY listadecamposdegrupo]
```

### 13.2. Uma instrução SELECT contendo uma cláusula GROUP BY possui as partes a seguir:

Parte	Descrição
<code>listadecampos</code>	O nome do campo ou dos campos a serem recuperados juntamente com qualquer alias de nome de campo, funções agregadas SQL, atributos de seleção (ALL, DISTINCT, DISTINCTROW ou TOP) ou outras opções da instrução SELECT.
<code>tabela</code>	O nome da tabela a partir da qual os registros são recuperados. Para obter maiores informações, consulte a cláusula FROM.
<code>critérios</code>	Critérios de seleção. Se a instrução incluir uma cláusula WHERE, o mecanismo de banco de dados Microsoft Jet agrupará valores depois de aplicar as condições WHERE aos registros.



listadecamposdegrupo	Os nomes de até 10 campos utilizados para agrupar os registros. A ordem dos nomes de campo em listadecamposdegrupo determina os níveis de agrupamento do nível mais alto ao mais baixo do agrupamento.
----------------------	--

### 13.3. Comentários

GROUP BY é opcional.

Os valores de resumo são omitidos se não houver uma função agregada SQL na instrução SELECT.

Valores Null nos campos GROUP BY são agrupados e não são omitidos. Contudo, valores Null não são avaliados em nenhuma função SQL agregada.

Utilize a cláusula WHERE para excluir linhas que você não deseja que permaneçam agrupadas e utilize a cláusula HAVING para filtrar os registros depois de eles terem sido agrupados.

Todos os campos na lista de campos SELECT devem estar incluídos na cláusula GROUP BY ou serem incluídos como argumentos em uma função SQL agregada.

## 14. Cláusula HAVING

Especifica quais registros agrupados são exibidos na instrução SELECT com uma cláusula GROUP BY. Depois de GROUP BY combinar os registros, HAVING exibirá qualquer registro agrupado pela cláusula GROUP BY que satisfaça às condições da cláusula HAVING.

### 14.1. Sintaxe

```
SELECT listadecampos
    FROM tabela
    WHERE critériosdeseleção
    GROUP BY listadecamposdegrupo
    [HAVING critériosdegrupo]
```

### 14.2. Uma instrução SELECT contendo uma cláusula HAVING possui as partes a seguir:

Parte	Descrição
listadecampos	O nome do campo ou campos a serem recuperados juntamente com qualquer alias de nome de campo, funções SQL agregadas, atributos de seleção (ALL, DISTINCT, DISTINCTROW ou TOP) ou outras opções da instrução SELECT.
tabela	O nome da tabela a partir da qual os registros são recuperados. Para obter maiores informações, consulte a cláusula FROM.
critériosdeseleção	Critérios de seleção. Se a instrução inclui uma cláusula WHERE, o mecanismo de banco de dados Microsoft Jet agrupa valores depois de aplicar as condições WHERE aos registros.

listadecamposdegrupo	Os nomes de até 10 campos utilizados para agrupar registros. A ordem dos nomes de campo em listadecamposdegrupo determina os níveis de agrupamento do nível mais alto ao mais baixo de agrupamento.
critériosdegrupo	Uma expressão que determina quais registros agrupados exibir.

### 14.3. Comentários

HAVING é opcional.

HAVING é semelhante a WHERE, que determina quais registros são selecionados. Depois de os registros serem agrupados com o GROUP BY, HAVING determina os registros a serem exibidos:

```
SELECT CódigoDaCategoria, Sum(UnidadesNoEstoque)
FROM Produtos
GROUP BY CódigoDaCategoria
HAVING Sum(UnidadesNoEstoque) > 100 And Like "BOS*";
```

Uma cláusula HAVING pode conter até 40 expressões vinculadas por operadores lógicos, como And e Or.

## 15. Cláusula IN

Identifica tabelas em qualquer banco de dados externo ao qual o mecanismo de banco de dados Microsoft Jet pode se conectar, como um banco de dados dBASE ou Paradox ou um banco de dados Microsoft Jet externo.

### 15.1. Sintaxe

Para identificar uma tabela de destino:

```
[SELECT | INSERT] INTO destino IN
{ caminho | ["caminho" "tipo"] | [""] [tipo; DATABASE = caminho] }
```

Para identificar uma tabela de origem:

```
FROM expressãodetabela IN
{ caminho | ["caminho" "tipo"] | [""] [tipo; DATABASE = caminho] }
```

### 15.2. Uma instrução SELECT contendo uma cláusula IN possui as partes a seguir:

Parte	Descrição
destino	O nome da tabela externa à qual os dados são inseridos.

expressãodetabela	O nome da tabela ou tabelas a partir das quais os dados são recuperados. Este argumento pode ser um único nome de tabela, uma consulta salva ou uma composição resultante de um INNER JOIN, LEFT JOIN ou RIGHT JOIN.
caminho	O caminho completo para o diretório ou arquivo contendo tabela.
tipo	O nome do tipo de banco de dados utilizado para criar tabela se um banco de dados não for um banco de dados Microsoft Jet (por exemplo, dBASE III, dBASE IV, Paradox 3.x ou Paradox 4.x).

### 15.3. Comentários

Você pode utilizar IN para se conectar a um único banco de dados externo de cada vez.

Em alguns casos, o argumento caminho refere-se ao diretório que contém os arquivos do banco de dados. Por exemplo, ao trabalhar com tabelas de banco de dados dBASE, FoxPro ou Paradox, o argumento caminho especifica o diretório contendo os arquivos .dbf ou .db. O nome do arquivo de tabela deriva do argumento destino ou expressãodetabela.

Para especificar um banco de dados não-Microsoft Jet, anexe um ponto-e-vírgula (;) ao nome e coloque-o entre aspas simples ( ' ') ou duplas ( " "). Por exemplo, tanto 'dBASE IV;' quanto "dBASE IV;" são aceitos.

Você também pode utilizar a palavra reservada DATABASE para especificar o banco de dados externo. Por exemplo, as linhas a seguir especificam a mesma tabela:

```
... FROM Table IN "" [dBASE IV; DATABASE=C:\DBASE\DATA\SALES;];
... FROM Table IN "C:\DBASE\DATA\SALES" "dBASE IV;"
```

Para um melhor desempenho e facilidade de uso, utilize uma tabela vinculada em lugar de IN.

Você também pode utilizar a palavra reservada IN como um operador de comparação em uma expressão. Para obter maiores informações, consulte o operador In.

## 16. Cláusula ORDER BY

Classifica os registros resultantes de uma consulta em um campo ou campos especificados, em ordem crescente ou decrescente.

### 16.1. Sintaxe

```
SELECT listadecampos
FROM tabela
WHERE critériosdeseleção
[ORDER BY campo1 [ASC | DESC ][, campo2 [ASC | DESC ]][, ...]]
```

### 16.2. Uma instrução SELECT contendo uma cláusula ORDER BY possui as partes a seguir:

Parte	Descrição
listadecampos	O nome do campo ou campos a serem recuperados juntamente com qualquer alias de nome de campo, funções SQL agregadas, atributos de seleção (ALL, DISTINCT, DISTINCTROW ou TOP) ou outras opções da instrução SELECT.

tabela O nome da tabela a partir da qual os registros são recuperados. Para obter maiores informações, consulte a cláusula FROM.

critériosde seleção Critérios de seleção. Se a instrução incluir uma cláusula WHERE, o mecanismo de banco de dados Microsoft Jet ordenará os valores depois de aplicar as condições WHERE aos registros.

campo1, campo2 Os nomes dos campos nos quais os registros devem ser classificados.

### 16.3. Comentários

ORDER BY é opcional. Entretanto, se desejar que os dados sejam exibidos em ordem classificada, então você deverá utilizar ORDER BY.

A ordem de classificação padrão é ascendente (de A a Z, de 0 a 9). Os dois exemplos a seguir classificam os nomes dos funcionários pela ordem do sobrenome:

```
SELECT Sobrenome, Nome
FROM Funcionários
ORDER BY Sobrenome;
```

```
SELECT Sobrenome, Nome
FROM Funcionários
ORDER BY Sobrenome ASC;
```

Para classificar em ordem decrescente (de Z a A, de 9 a 0), adicione a palavra reservada DESC ao final de cada campo que você queira classificar em ordem decrescente. O exemplo a seguir seleciona os salários e os classifica em ordem decrescente:

```
SELECT Sobrenome, Salário
FROM Funcionários
ORDER BY Salário DESC, Sobrenome;
```

Se você especificar um campo contendo dados de Memorando ou Objeto OLE na cláusula ORDER BY, um erro será gerado. O mecanismo de banco de dados Microsoft Jet não classifica campos desses tipos.

ORDER BY é geralmente o último item em uma instrução SQL.

Você pode incluir campos adicionais na cláusula ORDER BY. Os registros são classificados pelo primeiro campo listado após ORDER BY. Os registros que têm valores iguais naquele campo serão então classificados pelo valor no segundo campo listado e assim por diante.

## 17. Cláusula WHERE

Especifica quais registros das tabelas listadas na cláusula FROM são afetados por uma instrução SELECT, UPDATE ou DELETE.

### 17.1. Sintaxe

```
SELECT listadecampos
      FROM expressãodetabela
      WHERE critérios
```

### 17.2. Uma instrução SELECT contendo uma cláusula WHERE possui as partes a seguir:

Parte	Descrição
listadecampos	O nome do campo ou campos a serem recuperados juntamente com quaisquer aliases de nome de campo, atributos de seleção (ALL, DISTINCT, DISTINCTROW ou TOP) ou outras opções da instrução SELECT.
expressãodetabela	O nome da tabela ou tabelas a partir das quais os dados são recuperados.
critérios	Uma expressão que os registros devem atender para serem incluídos nos resultados da consulta.

### 17.3. Comentários

O mecanismo de banco de dados Microsoft Jet seleciona os registros que satisfazem às condições listadas na cláusula WHERE. Se você não especificar uma cláusula WHERE, a consulta retornará todas as linhas da tabela. Se você especificar mais de uma tabela na consulta e não tiver incluído uma cláusula WHERE ou uma cláusula JOIN, a consulta irá gerar um produto cartesiano das tabelas.

WHERE é opcional, mas quando incluído, se segue ao FROM. Por exemplo, você pode selecionar todos os funcionários no departamento de vendas (WHERE Depto = 'Vendas') ou todos os clientes com idades entre 18 e 30 anos (WHERE Idade Between 18 And 30).

Se você não utilizar uma cláusula JOIN para executar operações de associação SQL em várias tabelas, o objeto Recordset resultante não será atualizável.

WHERE é semelhante a HAVING. WHERE determina quais registros são selecionados. Da mesma forma, depois de os registros serem agrupados com GROUP BY, HAVING determina quais registros serão exibidos.

Utilize a cláusula WHERE para eliminar os registros que você não deseja que sejam agrupados por uma cláusula GROUP BY.

Utilize várias expressões para determinar quais registros a instrução SQL retorna. Por exemplo, a instrução SQL a seguir seleciona todos os funcionários cujos salários sejam maiores que R\$21.000:

```
SELECT Sobrenome, Salário
      FROM Funcionários
      WHERE Salário > 21000;
```

Uma cláusula WHERE pode conter até 40 expressões vinculadas por operadores lógicos, como And e Or.

Ao digitar um nome de campo que contém um espaço ou pontuação, coloque o nome entre colchetes ([ ]). Por exemplo, uma tabela de informações sobre o cliente poderia incluir informações sobre clientes específicos:

```
SELECT [Restaurante Favorito do Cliente]
```

Quando você especifica o argumento critérios, os literais de data deverão estar no formato dos EUA, mesmo que você não esteja utilizando a versão norte-americana do mecanismo de banco de dados Microsoft Jet. Por exemplo, 10 de maio de 1996 é escrito 10/5/96 no Brasil e 5/10/96 nos Estados Unidos. Certifique-se de colocar os literais de data entre sinais de número (#), como mostrado nos exemplos a seguir.

Para encontrar registros datados de 10 de maio de 1996 em um banco de dados dos Estados Unidos, você deve utilizar a instrução SQL a seguir:

```
SELECT *
      FROM Pedidos
      WHERE DataDeEnvio = #5/10/96#;
```

Você também pode utilizar a função DateValue, que é sensível às configurações internacionais estabelecidas pelo Microsoft Windows. Por exemplo, utilize este código para os Estados Unidos:

```
SELECT *
      FROM Pedidos
      WHERE DataDeEnvio = DateValue('5/10/96');
```

E utilize este código para o Brasil:

```
SELECT *
      FROM Pedidos
      WHERE DataDeEnvio = DateValue('10/5/96');
```

Se a coluna referenciada na sequência de critérios for do tipo GUID, a expressão de critérios utilizará uma sintaxe ligeiramente diferente:

```
WHERE ReplicaID = {GUID {12345678-90AB-CDEF-1234-567890ABCDEF}}
```

Não deixe de incluir as chaves e hifens embutidos, como mostrado.

## 18. Declaração WITH OWNERACCESS OPTION

Em um ambiente multiusuários com um grupo de trabalho seguro, use esta declaração com uma consulta para conceder ao usuário que executa a consulta as mesmas permissões que as do proprietário da consulta.

### 18.1. Sintaxe

instruçõesql WITH OWNERACCESS OPTION

### 18.2. Comentários

A declaração WITH OWNERACCESS OPTION é opcional.

O exemplo a seguir permite que o usuário visualize informações sobre salários (mesmo que, de outra forma, o usuário não tenha permissão para visualizar a tabela FolhaDePagamento), desde que o proprietário da consulta tenha essa permissão:

```
SELECT Sobrenome, Nome, Salário  
FROM Funcionários  
ORDER BY Sobrenome  
WITH OWNERACCESS OPTION;
```

Se de alguma outra forma o usuário for impedido de criar ou adicionar a uma tabela, você pode usar WITH OWNERACCESS OPTION para permitir que ele execute uma consulta criar tabela ou consulta acréscimo.

Se você deseja impor configurações de segurança de grupo de trabalho e permissões de usuários, não inclua a declaração WITH OWNERACCESS OPTION.

Essa opção requer que você tenha acesso ao arquivo System.mdw associado ao banco de dados. É realmente útil somente nas implementações de multiusuários com segurança.

## 19. Funções agregadas SQL

Utilizando os SQL funções agregadas, você pode determinar várias estatísticas em conjuntos de valores. Você pode utilizar estas funções em uma consulta e em expressões agregadas na propriedade SQL de um objeto QueryDef ou ao criar um objeto Recordset baseado em uma consulta SQL.

Funções Agregadas

Função Avg  
Função Count

Funções Min, Max  
Funções StDev, StDevP  
Função Sum  
Funções Var, VarP

## 20. Função Avg

Calcula a média aritmética de um conjunto de valores contido em um campo especificado em uma consulta.

### 20.1. Sintaxe

Avg(expr)

O espaço reservado expr representa uma expressão de sequência que identifica o campo que contém os dados numéricos dos quais você quer tirar uma média ou uma expressão que realiza um cálculo utilizando os dados naquele campo. Os operandos em expr podem incluir o nome de um campo de tabela, uma constante ou uma função (que pode ser intrínseca ou definida pelo usuário, mas nenhuma das outras funções agregadas SQL).

### 20.2. Comentários

A média calculada por Avg é a média aritmética (a soma dos valores dividida pelo número de valores). Você poderia utilizar o Avg, por exemplo, para calcular o custo médio do frete.

A função Avg não inclui nenhum campo Null no cálculo.

Você pode utilizar Avg em uma expressão de consulta e na propriedade SQL de um objeto QueryDef ou ao criar um objeto Recordset baseado em uma consulta SQL.

## 21. Função Count

Calcula o número de registros retornado por uma consulta.

### 21.1. Sintaxe

Count(expr)

O espaço reservado expr representa uma expressão de sequência de caracteres que identifica o campo que contém os dados que você quer contar ou uma expressão que realiza um cálculo utilizando os dados no campo. Os operandos em expr podem incluir o nome de um campo de tabela ou função (que pode ser intrínseca ou definida pelo usuário, mas nenhuma outra função agregada SQL). Você pode contar qualquer tipo de dados, inclusive texto.



## 21.2. Comentários

Você pode utilizar Count para contar o número de registros em uma consulta base. Por exemplo, você poderia utilizar Count para contar o número de pedidos enviados a um determinado país.

Embora expr possa realizar um cálculo em um campo, Count simplesmente computa o número de registros, independentemente dos valores armazenados nos registros.

A função Count não conta registros que tenham campos Null, exceto quando expr for o caractere curinga asterisco (\*). Se você utilizar um asterisco, Count calculará o número total de registro, inclusive aqueles que contêm campos Null. Count(\*) é consideravelmente mais rápido que Count([Nome da Coluna]). Não coloque o asterisco entre aspas ( ' '). O exemplo a seguir calcula o número de registros na tabela Pedidos:

```
SELECT Count(*) AS TotalDePedidos
FROM Pedidos;
```

Se expr identificar vários campos, a função Count contará um registro somente se um dos campos não for Null. Se todos os campos especificados forem Null, o registro não será contado. Separe os nomes de campo com um e-comercial (&). O exemplo a seguir mostra como você poderia limitar a contagem aos registros nos quais DataDeEnvio ou Frete não fosse Null:

```
SELECT Count('DataDeEnvio & Frete') AS [Not Null]
FROM Pedidos;
```

Você pode utilizar Count em uma expressão de consulta. Você também pode utilizar esta expressão na propriedade SQL de um objeto QueryDef ou durante a criação de um objeto Recordset baseado em uma consulta SQL.

## 22. Função Sum

Retorna a soma de um conjunto de valores contido em um campo especificado em uma consulta.

### 22.1. Sintaxe

Sum(expr)

O espaço reservado expr representa uma expressão de seqüência que identifica o campo que contém os dados numéricos que você quer adicionar ou uma expressão que realiza um cálculo utilizando os dados naquele campo. Os operandos em expr podem incluir o nome de um campo de tabela, uma constante ou uma função (que pode ser intrínseca ou definida pelo usuário, mas nenhuma das outras funções agregadas SQL).

### 22.2. Comentários

A função Sum totaliza os valores em um campo. Por exemplo, você poderia utilizar a função Sum para determinar o custo total dos encargos de frete.

A função Sum ignora os registros que contenham campos Null. O exemplo a seguir mostra como você pode calcular a soma dos produtos dos campos PreçoUnitário e Quantidade:

```
SELECT Sum(PreçoUnitário * Quantidade)AS [Receita Total]  
FROM [Detalhes do Pedido];
```

Você pode utilizar a função Sum em uma expressão de consulta. Você também pode utilizar esta expressão na propriedade SQL de um objeto QueryDef ou durante a criação de um objeto Recordset, com base em uma consulta SQL.

## 23. Operação INNER JOIN

Combina registros de duas tabelas sempre que houver valores correspondentes em um campo comum.

### 23.1. Sintaxe

```
FROM tabela1 INNER JOIN tabela2 ON tabela1.campo1 opercomp tabela2.campo2
```

### 23.2. A operação INNER JOIN possui as partes a seguir:

Parte	Descrição
tabela1, tabela2	Os nomes das tabelas das quais os registros são combinados.

campo1, campo2 Os nomes dos campos que são associados. Se não forem numéricos, os campos deverão ser do mesmo tipo de dados e conter o mesmo tipo de dados, mas não precisarão ter o mesmo nome.

opercomp Qualquer operador de comparação relacional: "=", "<," ">," "<=," ">=," ou "<>."

### 23.3. Comentários

Você pode usar uma operação INNER JOIN em qualquer cláusula FROM. Este é o tipo mais comum de associação. As associações internas combinam registros de duas tabelas sempre que houver valores correspondentes em um campo comum a ambas.

Você pode usar INNER JOIN com as tabelas Departamentos e Funcionários para selecionar todos os funcionários em cada departamento. Em contrapartida, para selecionar todos os departamentos (mesmo que alguns não tenham funcionários designados para eles) ou todos os funcionários (mesmo que alguns não sejam designados a um departamento), você pode usar uma operação LEFT JOIN ou RIGHT JOIN para criar uma associação externa.

Se você tentar associar campos contendo dados de Memorando ou Objeto OLE, ocorrerá um erro.

Você pode associar quaisquer dois campos numéricos de tipos semelhantes. Por exemplo, você pode associar em campos AutoNumeração e Longo pois são tipos semelhantes. Entretanto, você não pode associar os tipos de campo Único e Duplo.

O exemplo a seguir mostra como você poderia associar as tabelas Categorias e Produtos no campo CódigoDaCategoria:

```
SELECT NomeDaCategoria, NomeDoProduto
FROM Categorias
INNER JOIN Produtos
ON Categorias.CódigoDaCategoria = Produtos.CódigoDaCategoria;
```

No exemplo acima, CódigoDaCategoria é o campo associado, mas não é incluído na saída da consulta pois não está incluído na instrução SELECT. Para incluir o campo associado, inclua o nome do campo na instrução SELECT ¾ neste caso, Categorias.CódigoDaCategoria.

Você pode também vincular várias cláusulas ON em uma instrução JOIN, usando a sintaxe a seguir:

```
SELECT campos
FROM tabela1 INNER JOIN tabela2
ON tabela1.campo1 opercomp tabela2.campo1 AND
ON tabela1.campo2 opercomp tabela2.campo2) OR
ON tabela1.campo3 opercomp tabela2.campo3)];
```

Você pode também aninhar instruções JOIN usando a seguinte sintaxe:

```
SELECT campos
FROM tabela
INNER JOIN (tabela2 INNER JOIN [( ]tabela3
[INNER JOIN [( ]tabelax [INNER JOIN ...])
ON tabela3.campo3 opercomp tabelax.campo3])
ON tabela2.campo2 opercomp tabela3.campo3)
ON tabela1.campo1 opercomp tabela2.campo2;
```

Uma instrução LEFT JOIN ou RIGHT JOIN pode ser aninhada dentro de uma INNER JOIN, mas uma INNER JOIN não pode ser aninhada dentro de uma LEFT JOIN ou de uma RIGHT JOIN.

## 24. Operações LEFT JOIN, RIGHT JOIN

Combina registros da tabela de origem quando usados em qualquer cláusula FROM.

### 24.1. Sintaxe

```
FROM tabela1 [ LEFT | RIGHT ] JOIN tabela2
ON tabela1.campo1 opercomp tabela2.campo2
```

### 24.2. As operações LEFT JOIN e RIGHT JOIN possuem as partes a seguir:

Parte	Descrição
tabela1, tabela2	Os nomes das tabelas das quais os registros são combinados.

campo1, campo2	Os nomes dos campos que são associados. Os campos devem ser do mesmo tipo de dados e conter o mesmo tipo de dados, mas não precisam ter o mesmo nome.
opercomp	Qualquer operador de comparação relacional: "=", "<," ">," "<=," ">=," ou "<>."

### 24.3. Comentários

Utilize uma operação LEFT JOIN para criar uma associação externa esquerda. As associações externas esquerdas incluem todos os registros da primeira (esquerda) de duas tabelas, mesmo que não haja valores correspondentes para os registros na segunda tabela (direita).

Utilize uma operação RIGHT JOIN para criar uma associação externa direita. As associações externas direitas incluem todos os registros da segunda (direita) de duas tabelas, mesmo que não haja valores correspondentes para registros na primeira (esquerda) tabela.

Por exemplo, você poderia usar LEFT JOIN com as tabelas Departamentos (esquerda) e Funcionários (direita) para selecionar todos os departamentos, inclusive os que não tenham funcionários designados para eles. Para selecionar todos os funcionários, inclusive os que não estão designados para um departamento, você usaria RIGHT JOIN.

O exemplo a seguir mostra como você poderia associar as tabelas Categorias e Produtos no campo CódigoDaCategoria. A consulta produz uma lista de todas as categorias, inclusive aquelas que não contêm produtos:

```
SELECT NomeDaCategoria, NomeDoProduto
      FROM Categorias
      LEFT JOIN Produtos
      ON Categorias.CódigoDaCategoria = Produtos.CódigoDaCategoria;
```

Neste exemplo, CódigoDaCategoria é o campo associado, mas não é incluído nos resultados da consulta pois não está incluído na instrução SELECT. Para incluir o campo associado, digite o nome do campo na instrução SELECT ¾ neste caso, Categorias.CódigoDaCategoria.

Para criar uma consulta que inclua somente registros nos quais os dados nos campos associados sejam os mesmos, utilize uma operação INNER JOIN.

Uma operação LEFT JOIN ou RIGHT JOIN pode ser aninhada dentro de uma operação INNER JOIN, mas uma INNER JOIN não pode ser aninhada dentro de uma LEFT JOIN ou de uma RIGHT JOIN. Consulte a matéria sobre como aninhar no tópico INNER JOIN para ver como aninhar associações dentro de outras associações.

Você pode vincular várias cláusulas ON. Consulte a matéria sobre vinculação de cláusulas no tópico INNER JOIN para ver como isso é feito.

Se você tentar associar campos contendo dados de Memorando ou Objeto OLE, ocorrerá um erro.

## 25. Operação UNION

Cria uma consulta união, que combina os resultados de duas ou mais consultas ou tabelas independentes.

### 25.1. Sintaxe

[TABLE] consulta1 UNION [ALL] [TABLE] consulta2 [UNION [ALL] [TABLE] consultan [ ... ]]

### 25.2. A operação UNION possui as partes a seguir:

Parte	Descrição
consulta1-n	Uma instrução SELECT, o nome de uma consulta armazenada ou o nome de uma tabela armazenada precedida da palavra-chave TABLE.

Você pode mesclar os resultados de duas ou mais consultas, tabelas e instruções SELECT, em qualquer combinação, em uma única operação UNION. O exemplo a seguir mescla uma tabela existente denominada Novas Contas com uma instrução SELECT:

```
TABLE [Novas Contas] UNION ALL
SELECT *
FROM Clientes
WHERE QuantiaDoPedido > 1000;
```

Como padrão, nenhum registro duplicado é retornado quando você usa uma operação UNION; entretanto, você pode incluir o atributo ALL para assegurar que todos os registros sejam retornados. Isso faz com que a execução da consulta seja mais rápida.

Todas as consultas em uma operação UNION devem solicitar o mesmo número de campos; contudo, os campos não deverão ter o mesmo tamanho ou tipo de dados.

Use aliases somente na primeira instrução SELECT pois eles são ignorados em qualquer outra. Na cláusula ORDER BY, refira-se aos campos pelo que são chamados na primeira instrução SELECT.

Você pode usar uma cláusula GROUP BY ou HAVING em cada argumento consulta para agrupar os dados retornados.

Você pode usar uma cláusula ORDER BY no fim do último argumento consulta para exibir os dados retornados em uma ordem especificada.

## 26. Subconsultas SQL

Uma subconsulta é uma instrução SELECT aninhada dentro de uma instrução SELECT, SELECT...INTO, INSERT...INTO, DELETE ou UPDATE ou de uma outra subconsulta.

### 26.1. Sintaxe

Você pode usar três formas de sintaxe para criar uma subconsulta:

comparação [ANY | ALL | SOME] (instruçõesql)  
 expressão [NOT] IN (instruçõesql)  
 [NOT] EXISTS (instruçõesql)

### 26.2. Uma subconsulta possui as partes a seguir:

Parte	Descrição
comparação	Uma expressão e um operador de comparação que compara a expressão com os resultados da subconsulta.
expressão procurado. instruçõesql	Uma expressão para a qual o conjunto de resultados da subconsulta é procurado. Uma instrução SELECT, que segue o mesmo formato e regras de qualquer outra instrução SELECT. Deve estar entre parênteses.

### 26.3. Comentários

Você pode usar uma consulta em lugar de uma expressão na lista de campos de uma instrução SELECT ou em uma cláusula WHERE ou HAVING. Em uma subconsulta, você usa uma instrução SELECT para proporcionar um conjunto de um ou mais valores específicos para avaliar na expressão de cláusula WHERE ou HAVING.

Use o atributo ANY ou SOME, que são sinônimos, para recuperar registros na consulta principal que satisfaçam a comparação com qualquer registro recuperado na subconsulta. O exemplo a seguir retorna todos os produtos cujo preço unitário é maior que o de qualquer produto vendido com um desconto de 25% ou maior:

```
SELECT * FROM Produtos
WHERE PreçoUnitário > ANY
(SELECT PreçoUnitário FROM DetalhesDoPedido
WHERE Desconto >= .25);
```

Use o atributo ALL para recuperar somente os registros da consulta principal que satisfaçam a comparação com todos os registros recuperados na subconsulta. Se você tivesse alterado ANY para ALL no exemplo anterior, a consulta retornaria somente os produtos cujo preço unitário fosse maior que o de todos os produtos vendidos com um desconto de 25% ou maior. Isso é bem mais restritivo.

Use o atributo IN para recuperar somente os registros da consulta principal para os quais algum registro na subconsulta contenha um valor igual. O exemplo a seguir retorna todos os produtos com um desconto de 25% ou maior:

```
SELECT * FROM Produtos
WHERE CódigoDoProduto IN
(SELECT CódigoDoProduto FROM DetalhesDoPedido
WHERE Desconto >= .25);
```

Da mesma forma, você pode usar NOT IN para recuperar somente os registros na consulta principal para os quais nenhum registro na subconsulta contenha um valor igual.

Use o atributo EXISTS (com a palavra reservada NOT opcional) em comparações verdadeiro/falso para determinar se a subconsulta retorna algum registro.

Você pode também usar aliases de nome de tabela em uma consulta para se referir a tabelas relacionadas em uma cláusula FROM fora da subconsulta. O exemplo a seguir retorna os nomes dos funcionários cujos salários são iguais ou maiores do que o salário médio de todos os funcionários que têm o mesmo cargo. A tabela Funcionários recebe o alias "T1":

```
SELECT Sobrenome, Nome, Título, Salário
FROM Funcionários AS T1
WHERE Salário >= (SELECT Avg(Salário)
FROM Funcionários
WHERE T1.Título = Funcionários.Título) Order by Título;
```

No exemplo anterior, a palavra reservada AS é opcional.

Algumas subconsultas são permitidas em consultas de tabela de referência cruzada — especificamente, como atributos (os da cláusula WHERE). Subconsultas como saída (as da lista SELECT) não são permitidas em consultas de tabela de referência cruzada.

## 27. Instrução TRANSFORM

Cria uma consulta tabela de referência cruzada.

### 27.1. Sintaxe

```
TRANSFORM funçãoagrg
    instruçãoselect
    PIVOT campocentral [IN (valor1[, valor2[, ...]])]
```

### 27.2. A instrução TRANSFORM possui as partes a seguir:

Parte	Descrição
funçãoagrg	Uma função agregada SQL que opera sobre os dados selecionados.
instruçãoselect	Uma instrução SELECT.
campodinâmico	O campo ou expressão que você quer usar para criar títulos de coluna no conjunto de resultados da consulta.
valor1, valor2	Valores fixos usados para criar títulos de colunas.

### 27.3. Comentários

Quando resume dados utilizando uma consulta tabela de referência cruzada, você seleciona valores de campos ou expressões especificadas como títulos de colunas para poder visualizar os dados em um formato mais compacto do que com uma consulta seleção.

A instrução TRANSFORM é opcional, mas quando for incluída será a primeira instrução em uma sequência SQL. Ela antecede uma instrução SELECT que especifica os campos usados como títulos de linhas e uma cláusula GROUP BY que especifica o agrupamento de linhas. Opcionalmente, você pode incluir outras cláusulas, como WHERE, que especifiquem critérios adicionais de seleção ou de classificação. Você pode também usar subconsultas como atributos — especificamente, aqueles em uma cláusula WHERE — em uma consulta tabela de referência cruzada.

Os valores retornados em campodinâmico são usados como títulos de colunas no conjunto de resultados da consulta. Por exemplo, articular os números das vendas no mês da venda em uma consulta tabela de referência cruzada criaria 12 colunas. Você pode restringir campodinâmico para criar títulos a partir de valores fixos (valor1, valor2 ) relacionados na cláusula IN opcional. Você pode também incluir valores fixos para os quais não existam dados para criar colunas adicionais.

## 28. Declaração PARAMETERS

Declara o nome e o tipo de dados de cada parâmetro em uma consulta parâmetro.

### 28.1. Sintaxe

PARAMETERS nome tipodedados [, nome tipodedados [, ...]]

### 28.2. A declaração PARAMETERS possui as partes a seguir:

Parte	Descrição
nome	O nome do parâmetro. Atribuído à propriedade Name do objeto Parameter e usado para identificar esse parâmetro na coleção Parameters. Você pode usar nome como uma sequência de caracteres que é exibida em uma caixa de diálogo enquanto o aplicativo executa a consulta. Use colchetes ( [ ] ) para envolver o texto que contém espaços ou pontuação. Por exemplo, [Preço baixo] e [Começar relatório com que mês?] são argumentos nome válidos.
tipodedados	Um dos tipos de dados SQL do Microsoft Jet primários ou seus sinônimos.

### 28.3. Comentários

Para consultas executadas regularmente, você pode utilizar uma declaração PARAMETERS para criar uma consulta parâmetro. Uma consulta parâmetro pode ajudar a automatizar o processo de alteração dos critérios da consulta. Em uma consulta parâmetro, o código precisará fornecer os parâmetros a cada vez que a consulta for executada.



A declaração PARAMETERS é opcional, mas quando incluída precede qualquer outra instrução, inclusive SELECT.

Se a declaração incluir mais de um parâmetro, separe-os com vírgulas. O exemplo a seguir inclui dois parâmetros:

```
PARAMETERS [Preço baixo] Currency, [Data inicial] DateTime;
```

Você pode usar nome, mas não tipodedados em uma cláusula WHERE ou HAVING. O exemplo a seguir espera que dois parâmetros sejam fornecidos e, então, aplica os critérios aos registros na tabela Pedidos:

```
PARAMETERS [Preço baixo] Currency, [Data inicial] DateTime;  
SELECT NúmeroDoPedido, QuantiaDoPedido  
FROM Pedidos  
WHERE QuantiaDoPedido > [Preço baixo] AND DataDoPedido >= [Data inicial];
```

## 29. Operador Between...And

Determina se o valor de uma expressão se situa dentro de um intervalo especificado de valores. Você pode utilizar este operador em instruções SQL.

### 29.1. Sintaxe

```
expr [Not] Between valor1 And valor2
```

### 29.2. A sintaxe do operador Between...And possui as partes a seguir:

Parte	Descrição
expr	Expressão que identifica o campo que contém os dados a serem avaliados.
valor1, valor2	Expressões em relação as quais você deseja avaliar expr.

### 29.3. Comentários

Se o valor de expr estiver entre valor1 e valor2 (inclusive), o operador Between...And retornará True; caso contrário, retornará False. Você pode incluir o operador lógico Not para avaliar a condição oposta (isto é, se expr estiver situado fora do intervalo definido por valor1 e valor2).

Você poderia utilizar Between...And para determinar se o valor de um campo está situado em um intervalo numérico especificado. O exemplo a seguir determina se um pedido foi enviado a um local situado em um intervalo de códigos postais. Se o código postal estiver entre 98101 e 98199, a função Iif retornará "Local". Caso contrário, retornará "Nãolocal".

```
SELECT Iif(CódigoPostal Between 98101 And 98199, "Local", "Nãolocal") FROM Editores
```

Se expr, valor1 ou valor2 forem Null, Between...And retornará um valor Null.

Uma vez que os caracteres curinga, como \*, são tratados como literais, você não pode utilizá-los com o operador Between...And. Por exemplo, você não pode utilizar 980\* e 989\* para localizar todos os códigos postais que começam com 980 e 989. Em vez disso, você tem duas alternativas: pode adicionar uma expressão para a consulta que pegue os três caracteres da esquerda do campo de texto e utilizar Between...And nesses caracteres, ou pode preencher os valores superior e inferior com caracteres extras — neste caso, 98000 a 98999, ou 98000 a 98999 – 9999 se estiver utilizando códigos postais estendidos. (Você deve omitir o – 0000 dos valores inferiores pois, caso contrário, 98000 será excluído se alguns códigos postais tiverem seções e outros não.)

### 30. Operador In

Determina se o valor de uma expressão é igual a algum dos vários valores em uma lista especificada.

#### 30.1. Sintaxe

expr [Not] In(valor1, valor2, ...)

#### 30.2. A sintaxe do operador In possui as partes a seguir:

Parte	Descrição
expr	Expressão que identifica o campo que contém os dados a serem avaliados.
valor1, valor2	Expressão ou lista de expressões em relação a qual você deseja avaliar expr.

Se expr for encontrado na lista de valores, o operador In retornará True; caso contrário, retornará False. Você pode incluir o operador lógico Not para avaliar a condição oposta (isto é, se expr não está na lista de valores).

Por exemplo, você pode utilizar In para determinar os pedidos a serem enviados a um conjunto de regiões especificadas:

```
SELECT *  
FROM Pedidos  
WHERE RegiãoDeRemessa In ('Avon','Glos','Som')
```

### 31. Operador Like

Compara uma expressão de sequência com um padrão em uma expressão SQL.

#### 31.1. Sintaxe

expressão Like "padrão"

### 31.2.A sintaxe do operador Like possui as partes a seguir:

Parte	Descrição
Expressão	Expressão SQL utilizada em uma cláusula WHERE.
padrão	Seqüência ou literal de seqüência de caracteres em relação a qual expressão é comparada.

### 31.3.Comentários

Você pode utilizar o operador Like para localizar valores em um campo que correspondam ao padrão especificado. Para padrão, você pode especificar o valor completo (por exemplo, Like "Smith") ou pode utilizar caracteres curinga para encontrar um intervalo de valores (por exemplo, Like "Sm\*").

Em uma expressão, você pode utilizar o operador Like para comparar um valor de campo a uma expressão de seqüência. Por exemplo, se você digitar Like "C\*" em uma consulta SQL, a consulta retornará todos os valores de campo que começam com a letra C. Em uma consulta parâmetro, você pode solicitar ao usuário que forneça um padrão pelo qual procurar.

O exemplo a seguir retorna dados que começam com a letra P, seguida por qualquer letra entre A e F e três dígitos:

Like "P[A-F]###"

A tabela a seguir mostra como você pode utilizar Like para testar expressões para diferentes padrões.

Tipo de correspondência			
Padrão	Coincidente (retorna True)		Não coincidente (retorna False)
Vários caracteres	a*a	aa, aBa, aBBBa	aBC *ab* abc, AABb, Xab
Caractere especial	a[*]a	a*a	aaa
Vários caracteres	ab*	abcdeffg, abc	cab, aab
Um único caractere	a?a	aaa, a3a, aBa	aBBBa
Um único dígito	a#a	a0a, a1a, a2a	aaa, a10a
Intervalo de caracteres	[a-z]	f, p, j	2, &
Fora de um intervalo	[!a-z]	9, &, %	b, a
Nenhum dígito	[!0-9]	A, a, &, ~	0, 1, 9
Combinado	a[!b-m]#	An9, az0, a99	abc, aj0

### 32. CARACTERES CURINGA

A correspondência interna de padrões oferece uma ferramenta versátil para fazer comparações de seqüências. A tabela a seguir mostra os caracteres curinga que você pode utilizar com o operador Like e o número de dígitos ou seqüências aos quais eles podem corresponder.

Caractere(s)	
em padrão	
Coincide com expressão	
?	Qualquer caractere isolado
*	Zero ou mais caracteres
#	Qualquer dígito isolado (0 — 9)
[listadecaract]	Qualquer caractere isolado em listadecaract
[!listadecaract]	Qualquer caractere isolado não-presente em listadecaract

Você pode utilizar um grupo de um ou mais caracteres (listadecaract) entre colchetes ( [ ] ) para coincidir com qualquer caractere isolado em expressão, e listadecaract pode incluir praticamente qualquer caractere do conjunto de caracteres ANSI, inclusive dígitos. De fato, você pode utilizar os caracteres especiais colchete de abertura ( [ ), ponto de interrogação ( ? ), sinal de número ( # ) e asterisco ( \* ) para que correspondam diretamente a eles mesmos somente se estiverem entre colchetes. Você pode utilizar o colchete de fechamento ( ] ) dentro de um grupo para que corresponda a ele mesmo, mas pode utilizá-lo fora de um grupo como um caractere individual.

Além de uma simples lista de caracteres entre colchetes, listadecaract pode especificar um intervalo de caracteres através da utilização de um hífen ( - ) para separar os limites superior e inferior do intervalo. Por exemplo, a utilização de [A-Z] em padrão resultará em uma correspondência se a posição do caractere correspondente em expressão contiver qualquer uma das letras maiúsculas no intervalo de A a Z. Você pode incluir vários intervalos entre os colchetes sem delimitar os intervalos. Por exemplo, [a-zA-Z0-9] corresponde a qualquer caractere alfanumérico.

Outras regras importantes para correspondência de padrão incluem:

Um ponto de exclamação ( ! ) no início de listadecaract significa que uma correspondência será feita se qualquer caractere, exceto aqueles na listadecaract, for encontrado em expressão. Quando utilizado sem colchetes, o ponto de exclamação corresponderá a si mesmo.

Você pode utilizar o hífen ( - ) seja no início (depois de um ponto de exclamação, se este for utilizado) ou no fim de listadecaract para corresponder a si mesmo. Se estiver em qualquer outro local, o hífen identificará um intervalo de caracteres ANSI.

Quando você especifica um intervalo de caracteres, os caracteres devem aparecer em ordem de classificação crescente (A-Z ou 0-100). [A-Z] é um padrão válido, mas [Z-A] não é.

A seqüência de caracteres [ ] é ignorada; ela é considerada uma seqüência de comprimento zero ( "" ).

### 33. Instrução CREATE USER ou GROUP

Cria um ou mais novos usuários ou grupos.

#### Sintaxe

Criar um usuário:

```
CREATE USER senha de usuário pid [, senha de usuário pid, ...]
```

Criar um grupo:

```
CREATE GROUP grupo pid [, grupo pid, ...]
```

---

A instrução CREATE USER ou GROUP tem estas partes:

Parte	Descrição
<i>usuário</i>	O nome de um usuário a ser adicionado ao arquivo de informações do grupo de trabalho.
<i>Grupo</i>	O nome de um grupo a ser adicionado ao arquivo de informações do grupo de trabalho.
<i>Senha</i>	A senha a ser associada ao nome de <i>usuário</i> especificado.
<i>Pid</i>	A identificação pessoal.

#### Comentários

Um *usuário* e um *grupo* não podem ter o mesmo nome.

É necessária uma *senha* para cada *usuário* ou *grupo* criado.

### 34. Instrução ADD USER

Adiciona um ou mais *usuários* existentes a um *grupo* existente.

#### Sintaxe

```
ADD USER usuário [, usuário, ...] TO grupo
```

---

A instrução ADD USER tem estas partes:

Parte	Descrição
<i>Usuário</i>	O nome de um usuário a ser adicionado ao arquivo de informações do grupo de trabalho.
<i>Grupo</i>	O nome de um grupo a ser adicionado ao arquivo de informações do grupo de trabalho.

**Comentários**

Uma vez adicionado um *usuário* a um *grupo*, o *usuário* tem todas as permissões que foram concedidas ao *grupo*.

**35. Instrução DROP USER ou GROUP**

Exclui um ou mais *usuários* ou *grupos* existentes, ou remove um ou mais *usuários* existentes de um *grupo* existente.

**Sintaxe**

Excluir um ou mais *usuários* ou remover um ou mais *usuários* de um *grupo*:

DROP USER *usuário* [, *usuário*, ...] [FROM *grupo*]

Excluir um ou mais *grupos*:

DROP GROUP *grupo* [, *grupo*, ...]

---

A instrução DROP USER ou GROUP tem estas partes:

Parte	Descrição
<i>Usuário</i>	O nome de um usuário a ser removido do arquivo de informações do grupo de trabalho.
<i>Grupo</i>	O nome de um grupo a ser removido do arquivo de informações do grupo de trabalho..

**Comentários**

Se a palavra-chave FROM for usada na instrução DROP USER, cada um dos *usuários* listados na instrução será removido do *grupo* especificado em seguida à palavra-chave FROM. No entanto, os próprios *usuários* não serão excluídos.

A instrução DROP GROUP excluirá o(s) *grupo(s)* especificado(s). Os *usuários* que são membros do(s) *grupo(s)* não serão afetados, mas deixarão de ser membros do(s) *grupo(s)* excluído(s).

**36. Instrução ALTER USER ou DATABASE**

Altera a senha de um usuário existente ou de um banco de dados.

**Sintaxe**

ALTER DATABASE PASSWORD *senhanova senhaantiga*

ALTER USER *usuário* PASSWORD *senhanova senhaantiga*

---

A instrução ALTER USER ou DATABASE tem estas partes:

<b>Parte</b>	<b>Descrição</b>
<i>Usuário</i>	O nome de um usuário a ser adicionado ao arquivo de informações do grupo de trabalho.
<i>Senha nova</i>	A nova senha a ser associada ao nome de <i>usuário</i> ou <i>database</i> especificado.
<i>Senha antiga</i>	A senha existente a ser associada ao nome de <i>usuário</i> ou <i>grupo</i> especificado.

### 37. Instrução GRANT

Concede privilégios específicos a um usuário ou grupo existente.

#### Sintaxe

```
GRANT {privilegio[, privilegio, ...]} ON
      {TABLE tabela |
      OBJECT objeto |
```

```
CONTAINER recipiente } TO {authorizationname[, nomeautoriz, ...]}
```

---

A instrução GRANT tem estas partes:

<b>Parte</b>	<b>Descrição</b>
<i>Privilegio</i>	O privilégio ou privilégios a serem concedidos. Os privilégios são especificados usando as seguintes palavras-chave:  SELECT, DELETE, INSERT, UPDATE, DROP, SELECTSECURITY, UPDATESECURITY, DBPASSWORD, UPDATEIDENTITY, CREATE, SELECTSCHEMA, SCHEMA e UPDATEOWNER.
<i>Nome tabela</i> <i>Objeto</i>	Qualquer nome de tabela válido. Isto pode englobar qualquer objeto não-tabela. Uma consulta armazenada (modo de exibição ou procedimento) é um exemplo.
<i>Recipiente</i>	O nome de um recipiente válido.
<i>Nome autoriz</i>	O nome de um usuário ou grupo.

### 38. Instrução REVOKE

Revoga privilégios específicos de um usuário ou grupo existente.

#### Sintaxe

```
REVOKE {privilégio [, privilégio, ...]} ON  
  {TABLE tabela |  
  OBJECT objeto |  
  
  CONTAINER recipiente }  
FROM {nomeautoriz [, nomeautoriz, ...]}
```

---

A instrução REVOKE tem estas partes:

Parte	Descrição
<i>privilégio</i>	O privilégio ou privilégios a ser revogado. Os privilégios são especificados usando as seguintes palavras-chave:  SELECT, DELETE, INSERT, UPDATE, DROP, SELECTSECURITY, UPDATESECURITY, DBPASSWORD, UPDATEIDENTITY, CREATE, SELECTSCHEMA, SCHEMA e UPDATEOWNER.
<i>tabela</i>	Qualquer nome de tabela válido.
<i>objeto</i>	Isto pode englobar qualquer objeto não-tabela. Uma consulta armazenada ( <a href="#">modo de exibição</a> ou <a href="#">procedimento</a> ) é um exemplo.
<i>recipiente</i>	O nome de um recipiente válido.
<i>nomeautoriz</i>	O nome de um usuário ou grupo.



### 39. Tipos de dados SQL

Os tipos de dados [SQL](#) do [motor de banco de dados Microsoft Jet](#) consistem em 13 tipos de dados primários definidos pelo motor de banco de dados Microsoft® Jet e vários sinônimos válidos reconhecidos para esses tipos de dados.

A tabela a seguir lista os tipos de dados primários. Os sinônimos estão identificados em [Palavras reservadas SQL do motor de banco de dados Microsoft Jet](#).

Tipo de dados	Tamanho de armazenamento	Descrição
BINARY	1 byte por caractere	Qualquer tipo de dados pode ser armazenado em um campo deste tipo. Não é feita nenhuma conversão dos dados (por exemplo, para texto). O modo como os dados são inseridos em um campo binário dita o modo como eles aparecerão como saída.
BIT	1 byte	Valores e campos Yes e No que contenham somente um dos dois valores.
TINYINT	1 byte	Um valor inteiro entre 0 e 255.
MONEY	8 bytes	Um número inteiro escalado entre -922.337.203.685.477,5808 e 922.337.203.685.477,5807.
DATETIME (Consulte DOUBLE)	8 bytes	Um valor de data ou hora entre os anos 100 e 9999.
UNIQUEIDENTIFIER	128 bits	Um número de identificação exclusivo usado com chamadas de procedimento remoto.
REAL	4 bytes	Um valor de ponto flutuante de precisão simples com um intervalo entre - 3,402823E38 a - 1,401298E-45 para valores negativos, 1,401298E-45 a 3,402823E38 para valores positivos e 0.
FLOAT	8 bytes	Um valor de ponto flutuante de dupla precisão com um intervalo de - 1,79769313486232E308 a - 4,94065645841247E-324 para valores negativos, 4,94065645841247E-324 a 1.79769313486232E308 para valores positivos e 0.
SMALLINT	2 bytes	Um inteiro curto entre - 32.768 e 32.767.
INTEGER	4 bytes	Um inteiro longo entre - 2,147,483,648 e 2,147,483,647. (Consulte Observações)
DECIMAL	17 bytes	Um tipo de dados numéricos exatos que contenham valores no intervalo de 1028 - 1 a - 1028 - 1. Você pode definir a precisão (de 1 a 28) e a escala (0 - precisão definida). A precisão e escala padrões são 18 e 0, respectivamente.
TEXT	2 bytes por caractere (Consulte Observações)	Zero a um máximo de 2,14 gigabytes.
IMAGE	Conforme exigido	Zero a um máximo de 2,14 gigabytes. Usado para objetos OLE.

CHARACTER	2 bytes por caractere (Consulte Observações)	Zero a 255 caracteres.
-----------	---	------------------------

#### Observações

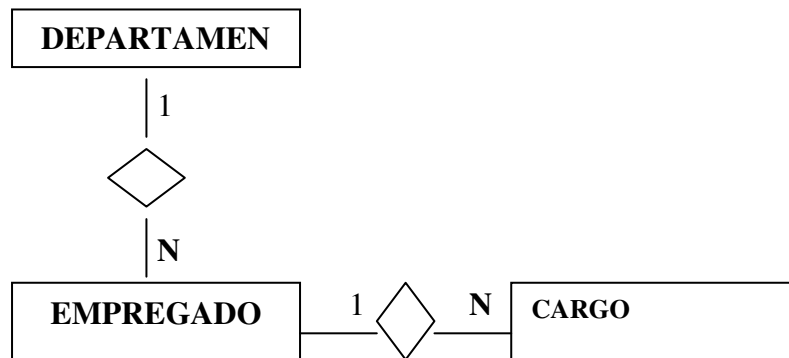
- A [semente](#) e o [incremento](#) podem ser modificados usando-se uma [instrução ALTER TABLE](#). Novas linhas inseridas na tabela terão valores, com base na nova semente e valores de incremento, que são gerados automaticamente para a coluna. Se a nova semente e incremento puderem produzir valores que correspondam a valores gerados com base na semente e incremento anteriores, serão geradas duplicações. Se a coluna for uma [chave primária](#), então inserir novas linhas pode resultar em erros quando são gerados valores duplicados.
- Para localizar o último valor usado em uma coluna de incrementação automática, você pode usar a instrução a seguir: `SELECT @@IDENTITY`. Você não pode especificar um nome de tabela. O valor retornado é da última tabela, contendo uma coluna de incrementação automática, que foi atualizada.
- Os caracteres em campos definidos como [TEXT](#) (também conhecidos como MEMO) ou [CHAR](#) (também conhecidos como TEXT(n) com um comprimento específico) são armazenados no [formato de representação Unicode](#). Os caracteres Unicode exigem uniformemente dois bytes para armazenar cada caractere. Para bancos de dados existentes do Microsoft Jet que contenham predominantemente dados de caracteres, isto pode significar que o arquivo do banco de dados quase dobraria de tamanho ao ser convertido para o formato do Microsoft Jet 4.0. No entanto, a representação Unicode de muitos conjuntos de caracteres, anteriormente identificados como Single-Byte Character Sets (SBCS, conjuntos de caracteres de byte único) pode ser facilmente compactada em um único byte. Para obter mais detalhes, consulte [CREATE TABLE](#). Se você definir uma coluna CHAR com o atributo COMPRESSION, os dados serão compactados automaticamente à medida que forem armazenados e descompactados quando recuperados a partir da coluna.



# EXERCÍCIOS SQL RESOLVIDOS



## SEGMENTO DE UM BANCO DE DADOS DE RECURSOS HUMANOS



Conteúdo das Tabelas

DEPARTAMENTO	Codigo	Nome
	01	Financeiro
	02	Engenharia
	03	Comercial

CARGO	Codigo	Nome	Salario
	01	Advogado	3000,00
	02	Analista de Sistemas	4000,00
	03	Contador	1000,00
	04	Engenheiro	4000,00
	05	Programador	1500,00
	06	Medico	4000,00
	07	Auxiliar de Escritório	400,00

EMPREGADO	Codigo	Nome	DataNascimento	CodDep	CodCargo
	001	Maria Araujo	10/10/70	01	03
	002	Carla Figueiredo	02/05/76	02	01
	003	Marcio Francisco	30/06/49	01	02
	004	Maria Bonita	20/07/80	03	05
	005	Francisco Carlos	18/08/72	02	02
	006	Rezende Rocha	25/03/60	01	01
	007	Marcia dos Anos	27/12/55	03	03
	008	José Ferreira	02/03/51	02	01
	009	Ferreira Fado	14/11/55	01	02


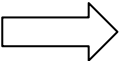
## 1. RECUPERANDO PARTE DOS DADOS DE UMA TABELA

Os dados recuperados em uma consulta são especificados entre a palavra **SELECT** e a palavra **FROM** de um comando **SELECT**.

### 1.1. Exemplo

Listar o código e o nome de cada empregado.

### 1.2. Solução

<b>SELECT</b>	Codigo, Nome		Atributos a serem listados
<b>FROM</b>	Empregado		Tabela onde estão os atributos a serem listados

### 1.3. Resultado

Codigo	Nome
001	Maria Araujo
002	Carla Figueiredo
003	Marcio Francisco
004	Maria Bonita
005	Francisco Carlos
006	Rezende Rocha
007	Marcia dos Anos
008	José Ferreira
009	Ferreira Fado

### 1.4. Observações

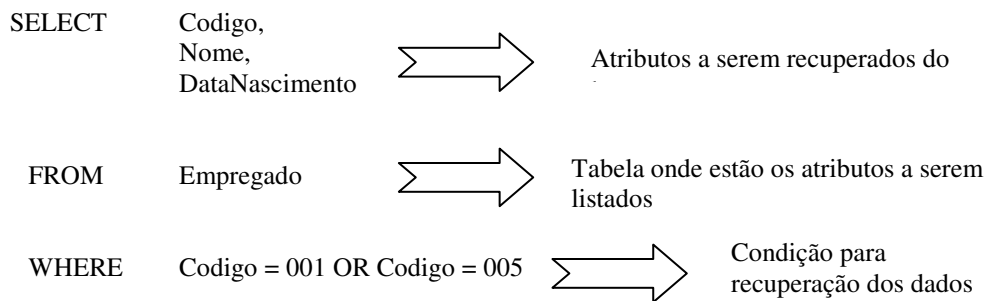
Observe que estão listados, apenas, os atributos especificados no comando **SELECT**, apesar da tabela conter outros atributos

## 2. RECUPERANDO DADOS DE UMA TABELA OBEDECENDO DETERMINADAS CONDIÇÕES.

As condições de recuperação de dados de uma ou mais tabelas são especificadas na cláusula WHERE do comando SELECT

2.1. Listar o código, o nome e a data de nascimento dos empregados de código 001 e 003.

2.2. Solução



2.3. Resultado

Codigo	Nome	DataNascimento
001	Maria Araujo	10/10/70
005	Francisco Carlos	18/08/72

2.4. Observações

- Repare que, apenas, os empregados cujos atributos satisfazem a condição especificada na cláusula WHERE foram listados.



### 3. LISTANDO TODOS OS ATRIBUTOS DE UMA TABELA.

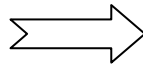
Para listar todos os dados de uma tabela basta especificar no lugar do nome dos atributos a serem listados um \* logo em seguida à palavra SELECT.

#### 3.1. Listar todos os atributos de cada cargo

#### 3.2. Solução

SELECT

\*



O asterisco indica que todos os atributos da tabela *Empregado* serão listados

FROM

Cargo

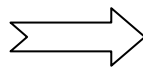


Tabela onde estão os atributos a serem listados

#### 3.3. Resultado

Codigo	Nome	Salario
01	Advogado	3000,00
02	Analista de Sistemas	4000,00
03	Contador	1000,00
04	Engenheiro	4000,00
05	Programador	1500,00

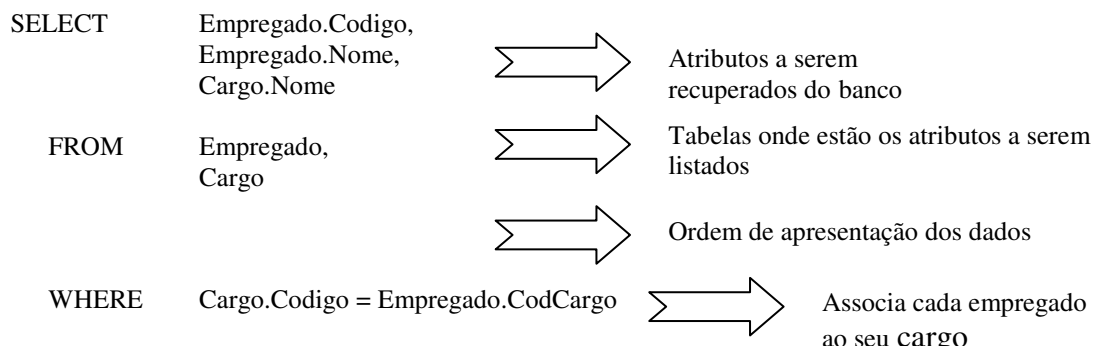
#### 4. RECUPERANDO DADOS DE MAIS DE UMA TABELA.

A recuperação de dados de mais de uma tabela requer a utilização da cláusula **WHERE** para associar uma tabela à outra, a menos que se queira fazer o produto cartesiano entre as tabelas considerados.

##### 4.1. Exemplo

Listar o código, o nome e o nome do cargo de cada empregado.

##### 4.2. Solução



##### 4.3. Resultado

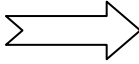
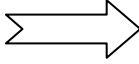
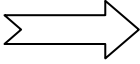
Codigo	Nome	CodCargo
001	Maria Araujo	Contador
002	Carla Figueiredo	Advogado
003	Marcio Francisco	Analista de Sistemas
004	Maria Bonita	Programador
005	Francisco Carlos	Analista de sistemas
006	Rezende Rocha	Advogado
007	Marcia dos Anos	Contador
008	José Ferreira	Advogado
009	Ferreira Fado	Analista de Sistema

## 5. RECUPERANDO DADOS DE MAIS DE UMA TABELA COM RESTRIÇÕES DE SELEÇÃO.

### 5.1. Exemplo

Listar o código, o nome e o nome do cargo de cada empregado. Somente os empregados vinculados aos departamentos 01 e 03 devem ser listados.

### 5.2. Solução

SELECT	Empregado.Codigo, Empregado.Nome, Cargo.Nome		Atributos a serem recuperados do banco
FROM	Empregado, Cargo		Tabelas onde estão os atributos a serem listados
WHERE	Cargo.Codigo = Empregado.CodCargo AND (Empregado.CodDep = 01 AND Empregado.CodDep = 03)		Associa cada empregado ao seu cargo e seleciona os departamentos 01 e 03 exigidos

### 5.3. Resultado

Codigo	Nome	CodCargo
001	Maria Araujo	Contador
003	Marcio Francisco	Analista de Sistemas
004	Maria Bonita	Programador
006	Rezende Rocha	Advogado
007	Marcia dos Anos	Contador
009	Ferreira Fado	Analista de Sistema

### 5.4. Observações

- Repare que a associação entre as tabelas e as condições de seleção dos dados aparecem todas na cláusula WHERE.

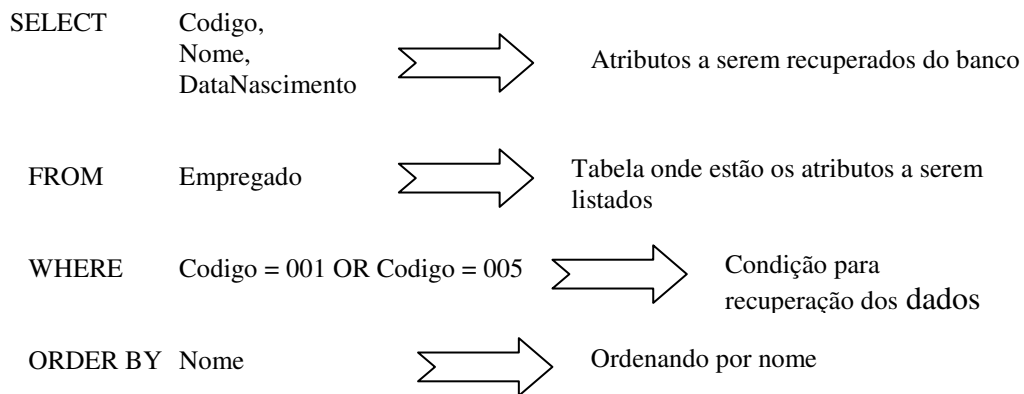
## 6. ORDENANDO DADOS RECUPERADOS.

Os dados recuperados podem ser ordenados de acordo com nossa necessidade. A ordenação dos dados é feita pela cláusula **SORT** do comando **SELECT**.

### 6.1. Exemplo

Listar o código, o nome e a data de nascimento dos empregados de código 001 e 003. Mostrar a lista em ordem crescente de nome do empregado.

### 6.2. Solução



### 6.3. Resultado

Codigo	Nome	DataNascimento
005	Francisco Carlos	18/08/72
001	Maria Araujo	10/10/70

### 6.4. Observações

- Repare que a lista está ordenada pela coluna Nome



## 8. RECUPERANDO DADOS DE MAIS DE DUAS TABELAS.

Listar o código e o nome do empregado, o código do departamento onde ele está vinculado e o nome do seu cargo e salário. A coluna correspondente ao nome do departamento deve Ter o título: *Departamento* e a correspondente ao nome do cargo: *Cargo*. Mostrar a lista em ordem decrescente de salário.

## 8.1. Solução

```

SELECT      Empregado.Codigo,
            Empregado.Nome,

            Departamento.Nome AS Departamento,
            Cargo.Nome AS Cargo,

            Cargo.Salario

FROM        Empregado,
            Departamento,
            Cargo

WHERE       Departamento.Codigo = Empregado.CodDep
            AND
            Empregado.CodCargo = Cargo.Codigo

ORDER BY    Cargo.Salario DESC

```

Mudando o título da coluna correspondente ao nome do departamento e do cargo

Tabela onde estão os atributos a serem listados

Associa o Empregado ao seu Departamento

Associa o Empregado ao seu Cargo

Ordenando por salario de modo decrescente

## 8.2. Resultado

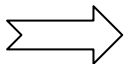
Codigo	Nome	Departamento	Cargo	Salario
003	Marcio Francisco	Financeiro	Analista de Sistemas	4000,00
005	Francisco Carlos	Engenharia	Analista de Sistemas	4000,00
009	Ferreira Fado	Financeiro	Analista de Sistemas	4000,00
002	Carla Figueiredo	Engenharia	Advogado	3000,00
006	Rezende Rocha	Financeiro	Advogado	3000,00
008	José Ferreira	Engenharia	Advogado	3000,00
004	Maria Bonita	Comercial	Programador	1500,00
001	Maria Araujo	Financeiro	Contador	1000,00
007	Marcia dos Anos	Comercial	Contador	1000,00

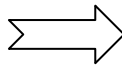
Repare os títulos das colunas Departamento e Cargo alterados e a ordem de apresentação da lista. Do maior salário para o menor.

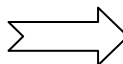
## 9. UTILIZANDO A FUNÇÃO DE AGREGAÇÃO: COUNT()

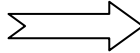
Listar a quantidade de empregados por cargo. A lista deve conter o código do cargo e a quantidade obtida. A coluna correspondente à quantidade de empregados deve ter o título: *Quantidade*.

### 9.1. Solução

SELECT      Empregado.CodCargo AS Cargo,            Mudando o título da coluna

                 COUNT(Empregado.Codigo) AS Quantidade            Contando os empregados e mudando o título da coluna

FROM          Empregado            Tabela onde estão os atributos a serem listados

GROUP BY    Empregado.CodCargo            Agrupando a quantidade de empregados por cargo

### 9.2. Resultado

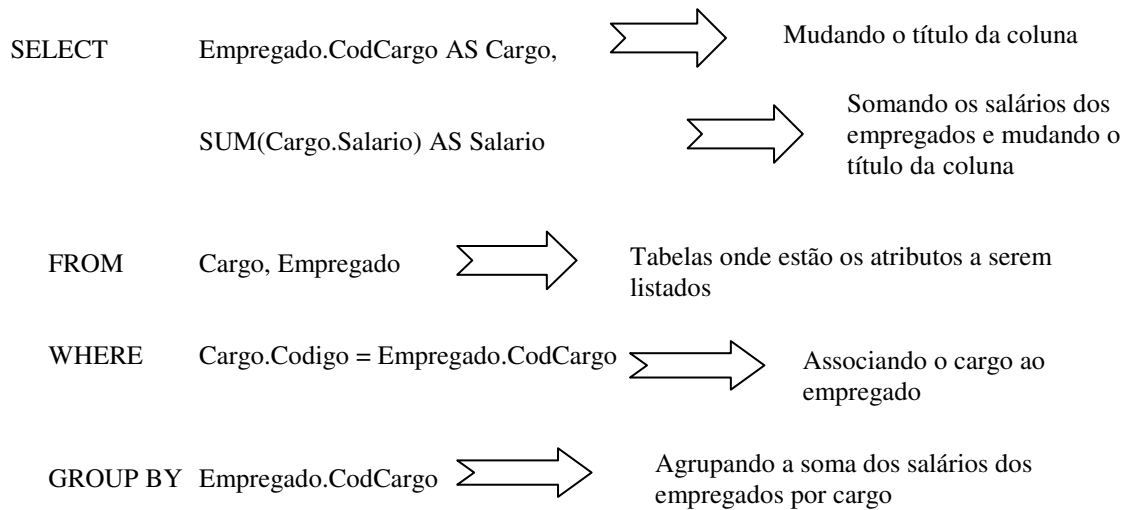
Cargo	Quantidade
01	03
02	03
03	02
05	01

### 9.3. Observações

- Repare os títulos das colunas. Recorra ao conteúdo da tabela Empregado para conferir o resultado da lista obtida.

**10. UTILIZANDO A FUNÇÃO DE AGREGAÇÃO: SUM()**

Listar a soma dos salários dos empregados de cada cargo. A lista deve conter o código do cargo e a soma dos salários obtida. A coluna correspondente à soma dos salários deve ter o título: *Salário*.

**10.1. Solução****10.2. Resultado**

Cargo	Salário
01	9.000,00
02	12.000,00
03	2.000,00
05	1.500,00

Repare os títulos das colunas. Recorra ao conteúdo das tabelas Empregado e Cargos para conferir o resultado da lista obtida.

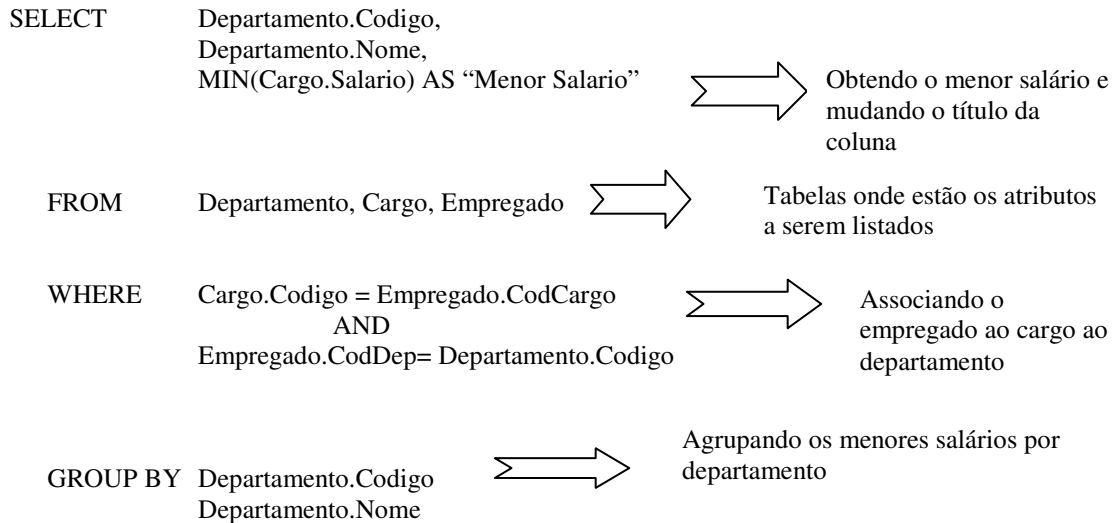
Observe que só foram listados os cargos para os quais tem algum empregado vinculado.



## 11. UTILIZANDO A FUNÇÃO DE AGREGAÇÃO: MIN()

Listar o menor salário de cada departamento. A lista deve conter o código, o nome e o menor salário do departamento. A coluna correspondente ao menor salário deve ter o título: *Menor Salário*.

### 11.1. Solução



### 11.2. Resultado

Codigo	Nome	Menor Salário
01	Financeiro	1.000,00
02	Engenharia	3.000,00
03	Comercial	1.000,00

### 11.3. Observações

- Repare os títulos das colunas. Recorra ao conteúdo das tabelas Departamento, Empregado e Cargos para conferir o resultado da lista obtida.
- Observe que só foram listados os departamentos para os quais tem algum empregado vinculado.
- Repare que as cláusulas FROM e WHERE fazem referência à tabela Empregado, embora nenhum dado desta tabela é pedido na lista. Ela aparece porque está no caminho dos relacionamentos que conduzem a tabela Departamento (que contém dados a serem listados) e a tabela Cargo (que contém dados a serem listados). Na realidade o menor salário diz respeito aos empregados vinculados aos departamentos.

## 12. UTILIZANDO A FUNÇÃO DE AGREGAÇÃO: MAX()

Listar o maior salário de cada departamento. A lista deve conter o código, o nome e o maior salário do departamento. A coluna correspondente ao maior salário deve ter o título: *Maiorr Salário*.

### 12.1. Solução

```

SELECT      Departamento.Codigo,

            Departamento.Nome,

            MAX(Cargo.Salario) AS "Maior Salario"
FROM        Departamento, Cargo, Empregado
WHERE       Cargo.Codigo = Empregado.CodCargo
            AND
            Empregado.CodDep= Departamento.Codigo
GROUP BY    Departamento.Codigo,
            Departamento.Nome
  
```

Obtendo o maior salário e mudando o título da coluna

Tabelas onde estão os atributos a serem listados

Associando o o empregado cargo e ao departamento

Agrupando o maior salário por departamento

### 12.2. Resultado

Codigo	Nome	Maior Salário
01	Financeiro	4.000,00
02	Engenharia	4.000,00
03	Comercial	1.500,00

### 12.3. Observações

- Repare os títulos das colunas. Recorra ao conteúdo das tabelas Departamento, Empregado e Cargos para conferir o resultado da lista obtida.
- Observe que só foram listados os departamentos para os quais tem algum empregado vinculado.

- Repare que as cláusulas FROM e WHERE fazem referência à tabela Empregado, embora nenhum dado desta tabela é pedido na lista. Ela aparece porque está no caminho dos relacionamentos que conduzem a tabela Departamento (que contém dados a serem listados) e a tabela Cargo (que contém dados a serem listados). Na realidade o menor salário diz respeito aos empregados vinculados aos departamentos.

### 13. UTILIZANDO A FUNÇÃO DE AGREGAÇÃO: AVG()

A cláusula AVG calcula a média aritmética dos valores de uma determinada coluna. Linhas cuja coluna tenha valor nulo não entra no cálculo da média.

#### 13.1. Exmplo

Listar o média dos salários de cada departamento. A lista deve conter o código, o nome e a média dos salários do departamento. A coluna correspondente à média dos salários deve ter o título: *Média dos Salários*.

#### 13.2. Solução

```

SELECT      Departamento.Codigo,

            Departamento.Nome,

            AVG(Cargo.Salario) AS "Média dos Salarios"
FROM        Departamento, Cargo, Empregado
WHERE       Cargo.Codigo = Empregado.CodCargo
            AND
            Empregado.CodDep= Departamento.Codigo

GROUP BY    Departamento.Codigo,
            Departamento.Nome
  
```

Obtendo média dos salários e mudando o título da coluna

Tabelas onde estão os atributos a serem listados

Associando o empregado ao cargo e ao departamento

Agrupando média dos salários por departamento

#### 13.3. Resultado

Codigo	Nome	Média dos Salarios
01	Financeiro	3.000,00
02	Engenharia	3.333,33
03	Comercial	1.250,00

#### 13.4. Observações

- Repare os títulos das colunas. Recorra ao conteúdo das tabelas Departamento, Empregado e Cargo para conferir o resultado da lista obtida.
- Observe que só foram listados os departamentos para os quais tem algum empregado vinculado.
- Repare que as cláusulas FROM e WHERE fazem referência à tabela Empregado, embora nenhum dado desta tabela é pedido na lista. Ela aparece porque está no caminho dos relacionamentos que conduzem a tabela Departamento (que contém dados a serem listados) e a tabela Cargo (que contém dados a serem listados). Na realidade o menor salário diz respeito aos empregados vinculados aos departamentos.

#### 14. UTILIZANDO A CLAUSULA: **HAVING()**

A clausula HAVING é utilizada para selecionar dados (filtrar) obtidos por uma função de agregação. Ela seleciona os dados depois que eles são agrupados, diferentemente da clausula WHERE que seleciona antes.

##### 14.1. Exemplo

Listar os departamentos cuja média dos salários sejam menor que R\$ 3.000,00. A lista deve conter o código, o nome e a média dos salários do departamento. A coluna correspondente à média dos salários deve ter o título: *Média dos Salários*.

##### 14.2. Solução

```

SELECT      Departamento.Codigo,

            Departamento.Nome,

            AVG(Cargo.Salario) AS "Menor Salario"
FROM        Departamento, Cargo, Empregado
WHERE       Cargo.Codigo = Empregado.CodCargo
            AND
            Empregado.CodDep= Departamento.Codigo
GROUP BY    Departamento.Codigo,
            Departamento.Nome
HAVIING     AVG(Cargo.Salario) < 3000,00
  
```

Obtendo a média dos salários e mudando o título da coluna

Tabelas onde estão os atributos a serem listados

Associando o o empregado cargo e ao departamento

Agrupando a média dos salários por departamento

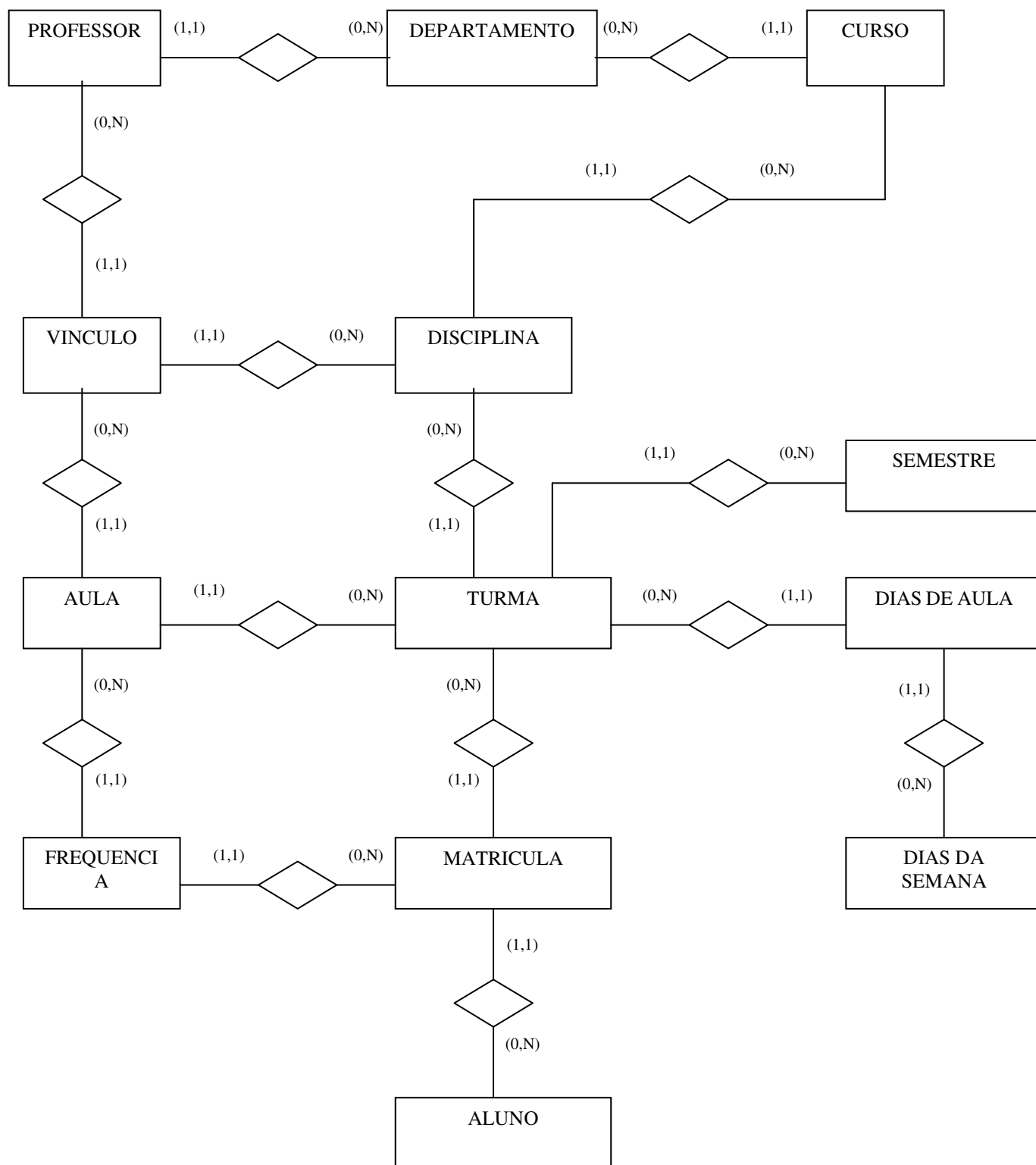
Selecionando as médias de salário menor que 3000,00

##### 14.3. Resultado

Codigo	Nome	Menor Salário
03	Comercial	1.500,00

#### 14.4. Observações

- Repare os títulos das colunas. Recorra ao conteúdo das tabelas Departamento, Empregado e Cargos para conferir o resultado da lista obtida.
- Observe que só foram listados os departamentos para os quais tem algum empregado vinculado.
- Repare que as cláusulas FROM e WHERE fazem referência à tabela Empregado, embora nenhum dado desta tabela é pedido na lista. Ela aparece porque está no caminho dos relacionamentos que conduzem a tabela Departamento (que contém dados a serem listados) e a tabela Cargo (que contém dados a serem listados). Na realidade o menor salário diz respeito aos empregados vinculados aos departamentos.

**O BANCO ACADÊMICO**



**Definição dos Atributos**

<b>PROFESSOR</b>	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	Cpf	S		S	N	11	0	
	Nome			S	C	30	0	
	Data de Nascimento			S	D			
	Logradouro			S	C	30	0	
	Bairro			S	C	20	0	
	Cidade			S	C	20	0	
	Estado			S	C	2	0	
	Cep			S	N	8	0	
	CodigoDepartamento		S	S				

<b>DEPARTAMENTO</b>	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	Código	S		S	C	3	0	
	Nome			S	C	30	0	

<b>CURSO</b>	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	Codigo	S		S	C	3	0	
	Nome			S	C	30	0	
	CodigoDepartamento		S	S				

<b>DISCIPLINA</b>	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	CodigoCurso	S	S	S				
	Codigo	S		S	N	4	0	
	Nome			S	C	30	0	
	Creditos			S	N	1		
	Preleção			S	N	1		
	Laboratorio			S	S	1		

<b>VINCULO</b>	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	CpfProfessor	S	S	S				
	CodigoCurso	S	S	S				
	CodigoDisciplina	S	S	S				

<b>DIA DA SEMANA</b>	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	Codigo	S		S	N	1	0	
	Nome			S	C	9	0	

DIAS DE AULA	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	CodigoCurso	S	S	S				
	CodigoDisciplina	S	S	S				
	NumeroTurma	S	S	S				
	NumeroSubTurma	S	S	S				
	AnoSemestre	S	S	S				
	NumeroSemestre	S	S	S				
	CodigoDiaSemana	S	S					
	TipoAula			S	C	1		
	Horario			S	C	2		

AULA	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	CodigoCurso	S	S	S				
	CodigoDisciplina	S	S	S				
	NumeroTurma	S	S	S				
	NumeroSubTurma	S	S	S				
	AnoSemestre	S	S	S				
	NumeroSemestre	S	S	S				
	Data	S		S	D			
	Numero	S		S	N	1		
	Resumo			S	C	60		
	CPFProfessor		S	S				

FREQUENCIA	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	CodigoCurso	S	S	S				
	CodigoDisciplina	S	S	S				
	NumeroTurma	S	S	S				
	NumeroSubTurma	S	S	S				
	AnoSemestre	S	S	S				
	NumeroSemestre	S	S	S				
	DataAula	S	S	S				
	NumeroAula	S	S	S				
	MatriculaAluno	S	S	S				
	Data da Matrícula	S	S	S				
	IndPresença			S	C	1		

MATRICULA	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	CodigoCurso	S	S	S				
	CodigoDisciplina	S	S	S				
	NumeroTurma	S	S	S				
	NumeroSubTurma	S	S	S				
	AnoSemestre	S	S	S				
	NumeroSemestre	S	S	S				
	MatriculaAluno	S	S	S				
	DataMatricula	S		S	D			
	Nota Final			S	N	3	1	

TURMA	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	CodigoCurso	S	S	S				
	CodigoDisciplina	S	S	S				
	Numero	S		S	C	3		
	NumeroSubTurma	S		S	C	1		
	AnoSemestre	S	S	S				
	NumeroSemestre	S	S	S				

ALUNO	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	Matricula	S		S	N	10	0	
	Nome			S	C	30	0	
	Data de Nascimento			S	D			
	Logradouro			S	C	30	0	
	Bairro			S	C	20	0	
	Cidade			S	C	20	0	
	Estado			S	C	2	0	
	Cep			S	N	8	0	
	Sexo		S	S				

SEMESTRE	ATRIBUTOS	ID	CE	OB	NAT	TAM	DEC	DOM
	Ano	S		S	N	4	0	
	Numero	S		S	N	1	0	
	Data de Inicio das Aulas			S	D			
	Data de Terminio das Aulas			S	D			

### Conteúdo das Tabelas

PROFESSOR	Cpf	Nome	Data de Nascimento	Logradouro	Bairro	Cidade	Estado	CEP	Codigo Departamento
	11111111111	Ivon Canedo	04/05/47	Rua 1	S Bueno	Goiânia	GO	74000-001	CMP
	22222222222	Marcio Kanutto	30/10/50	Rua 2	S Sul	Goiânia	GO	74000-002	CMP
	33333333333	Flácio dos Reis	10/03/43	Rua 2	Ipanema	Rio de Janeiro	RJ	60000-003	FIT
	44444444444	Maria Ribeiro	30/06/60	Rua 3	Pavuna	São Paulo	SP	50000-004	ADM
	55555555555	Carla Lemos	30/07/52	Rua 4	S Oeste	Goiânia	GO	74000-005	ADM
	66666666666	Marcos Cintra	20/04/51	Rua 5	S Central	Goiânia	GO	75000-006	ENG
	77777777777	Francisco Chagas	12/10/55	Rua 6	Vila Nova	B Horizonte	MG	45000-007	ENG
	88888888888	Virgílio	14/03/56	Rua 7	S Universitário	Salvador	BA	35000-008	BIO
	99999999999	Francisco Reis	20/10/58	Rua 8	Marista	Goiânia	GO	36000-009	FIt

DEPARTAMENTO	Codigo	Nome
	CMP	CIÊNCIA DA COMPUTAÇÃO
	FIT	FILOSOFIA E TEOLOGIA
	ADM	ADMINISTRAÇÃO
	ENG	ENGENHARIA
	ECO	ECONOMIA
	BIO	BIOLOGIA
	ZOO	ZOOTECNICA
	MAF	MATEMÁTICA E FÍSICA
	FAR	FARMÁCIA

<b>CURSO</b>	Codigo	Nome	Codigo Departamento
	CMP	CIÊNCIA DA COMPUTAÇÃO	CMP
	ENC	ENGENHARIA DE COMPUTAÇÃO	CMP
	FIL	FILOSOFIA E TEOLOGIA	FIT
	TEO	TEOLOGIA	FIT
	ADM	ADMINISTRAÇÃO	ADM
	ENG	ENGENHARIA CIVIL	ENG
	ECO	ECONOMIA	ECO
	BIO	BIOLOGIA	BIO
	ZOO	ZOOTECNICA	BIO
	MAT	MATEMÁTICA	MAF
	FIS	FISICA	MAF

<b>DISCIPLINA</b>	CodigoCurso	Codigo	Nome	Creditos	Preleção	Laborat ório
	CMP	4070	ENGENHARIA DE SOFTWARE	6	4	2
	CMP	4433	BANCO DE DADOS I	4	2	2
	CMP	4033	PROJETO DE SISTEMA	6	6	0
	CMP	3243	ANÁLISE	4	4	0
	ADM	1000	INTRODUÇÃO A ADMINISTRAÇÃO	4	4	0
	FIL	3444	INTRODUÇÃO A FILOSOFIA	4	4	0
	CMP	3030	BANCO DE DADOS II	6	4	2
	MAT	1010	CÁLCULO I	4	4	0
	FIS	1020	FISICA I	6	4	0
	ADM	4040	ADMINISTRAÇÃO PÚBLICA	4	4	0
	MAT	8080	ALGEBRA LINEAR	6	6	0
	FIS	2020	FISICA II	6	4	2
	FIL	9090	FILOFIA II	4	4	0

VINCULO	CpfProfessor	CodigoCurso	CodigoDisciplina
	11111111111	CMP	4070
	22222222222	CMP	4433
	22222222222	CMP	4033
	11111111111	CMP	3030
	33333333333	CMP	3243
	44444444444	ADM	1000
	33333333333	CMP	3030
	55555555555	ADM	4040
	66666666666	FIL	3444
	77777777777	FIS	1020
	77777777777	FIS	2020
	88888888888	MAT	8080
	88888888888	MAT	1010

DIA DA SEMANA	Codigo	Nome
	1	DOMINGO
	2	SEGUNDA-FEIRA
	3	TERÇA-FEIRA
	4	QUARTA-FEIRA
	5	QUINTA-FEIRA
	6	SEXTA-FEIRA
	7	SÁBADO

<b>TURMA</b>	Codigo Curso	Codigo Disciplina	Número	Numero SubTurma	Ano Semestre	Numero Semestre
	CMP	4070	A01	1	1999	2
	CMP	4070	A01	2	1999	2
	CMP	4070	A02	1	1999	2
	CMP	4070	A01	2	1999	2
	CMP	4433	A01	0	1999	2
	CMP	4433	C01	0	1999	2
	CMP	4433	A02	0	1999	2
	CMP	4033	A01	0	1999	2
	ADM	1000	A01	0	1999	2
	ADM	1000	C01	0	1999	2
	FIL	3444	A01	0	1999	2
	MAT	1010	A01	0	1999	2
	MAT	1010	C01	0	1999	2
	FIS	2020	A01	0	1999	2
	FIS	2020	C01	0	1999	2
	CMP	3243	A01	0	1999	2
	CMP	3243	C01	0	1999	2
	CMP	3243	A02	0	1999	2
	CMP	3030	A01	0	1999	2

<b>DIAS DE AULA</b>	<b>Codigo Curso</b>	<b>Codigo Disciplina</b>	<b>Numero Turma</b>	<b>Número Subturma</b>	<b>Ano Semestre</b>	<b>Numero Semestre</b>	<b>Codigo Dia Semana</b>	<b>Tipo Aula</b>	<b>Horário</b>
	CMP	4070	A01	0	1999	2	4	P	1M
	CMP	4070	A01	0	1999	2	7	P	1M
	CMP	4070	A01	1	1999	2	4	L	2M
	CMP	4070	A01	2	1999	2	7	L	3M
	CMP	4070	C01	0	1999	2	4	P	1N
	CMP	4070	C01	0	1999	2	6	P	1N
	CMP	4070	C01	1	1999	2	4	L	2N
	CMP	4070	C01	2	1999	2	6	L	3N
	CMP	4433	A01	0	1999	2	2	P	1M
	CMP	4433	A01	1	1999	2	5	L	1M
	CMP	4433	A01	2	1999	2	5	L	2M
	CMP	4433	C01	0	1999	2	3	P	2N
	CMP	4433	C01	1	1999	2	6	L	2N
	CMP	4433	C01	2	1999	2	6	L	3N
	CMP	4433	A02	0	1999	2	4	P	1M
	CMP	4433	A02	1	1999	2	7	L	1M
	CMP	4433	A02	2	1999	2	7	L	2M
	CMP	3030	A01	0	1999	2	2	P	2M
	CMP	3030	A01	0	1999	2	5	P	1M
	CMP	3030	A01	1	1999	2	5	L	2M
	CMP	3030	A01	2	1999	2	5	L	3M
	ADM	1000	A01	0	1999	2	4	P	1M
	ADM	1000	A01	0	1999	2	7	P	1M
	ADM	1000	C01	0	1999	2	4	P	2N
	ADM	1000	C01	0	1999	2	7	P	2N
	FIL	3444	A01	0	1999	2	2	P	1M
	FIL	3444	A01	0	1999	2	4	P	1M
	FIS	2020	A01	0	1999	2	5	P	1M
	FIS	2020	A01	0	1999	2	7	P	1M
	FIS	2020	A01	1	1999	2	7	L	2M
	FIS	2020	A01	2	1999	2	7	L	3M
	FIS	2020	C01	0	1999	2	3	P	1N



	FIS	2020	C01	0	1999	2	6	P	1N
	FIS	2020	C01	1	1999	2	6	L	2N
	FIS	2020	C01	2	1999	2	6	L	3N
	MAT	1010	A01	0	1999	2	4	P	3M
	MAT	1010	A01	0	1999	2	7	P	3M
	MAT	1010	C01	0	1999	2	4	P	3N
	MAT	1010	C01	0	1999	2	7	P	3N
	CMP	4033	C01	0	1999	2	3	P	3N
	CMP	4033	C01	0	1999	2	4	P	3N
	CMP	4033	C01	0	1999	2	7	P	3N

AULA	Codigo Curso	Codigo Disciplina	Numero Turma	Numero Subturma	Ano Semestre	Numero Semestre	Data	Número	Resumo	Cpf Professor
	CMP	4070	A01	0	1999	2	03/03/99	1	RESUMO 1	1111111111
	CMP	4070	A01	0	1999	2	06/03/99	1	RESUMO 2	1111111111
	CMP	4070	A01	1	1999	2	03/03/99	1	RESUMO 3	1111111111
	CMP	4070	A01	2	1999	2	06/03/99	2	RESUMO 3	1111111111
	CMP	4070	C01	0	1999	2	03/03/99	1	RESUMO 1	1111111111
	CMP	4070	C01	0	1999	2	05/03/99	1	RESUMO 2	1111111111
	CMP	4070	C01	1	1999	2	03/03/99	2	RESUMO 3	1111111111
	CMP	4070	C01	2	1999	2	05/03/99	2	RESUMO 3	1111111111
	CMP	4433	A01	0	1999	2	01/03/99	1	RESUMO 1	2222222222
	CMP	4433	A01	1	1999	2	04/03/99	1	RESUMO 2	2222222222
	CMP	4433	A01	2	1999	2	04/03/99	1	RESUMO 2	2222222222
	CMP	4433	C01	0	1999	2	02/03/99	1	RESUMO 1	2222222222
	CMP	4433	C01	1	1999	2	05/03/99	1	RESUMO 2	2222222222
	CMP	4333	C01	2	1999	2	05/03/99	2	RESUMO 2	2222222222
	CMP	4433	A02	0	1999	2	02/03/99	1	RESUMO 2	2222222222
	CMP	4433	A02	1	1999	2	06/03/99	1	RESUMO 3	3333333333
	CMP	4433	A02	2	1999	2	06/03/99	2	RESUMO 4	3333333333
	CMP	4033	C01	0	1999	2	02/03/99	1	RESUMO 1	2222222222
	CMP	4033	C01	0	1999	2	03/03/99	1	RESUMO 2	2222222222
	CMP	4033	C01	0	1999	2	06/03/99	1	RESUMO 3	2222222222
	CMP	3030	A01	0	1999	2	01/03/99	1	RESUMO 1	3333333333

	CMP	3030	A01	0	1999	2	04/03/99	1	RESUMO 2	3333333333
	CMP	3030	A01	1	1999	2	04/03/99	1	RESUMO 3	3333333333
	CPM	3030	A01	2	1999	2	04/03/99	1	RESUMO 3	3333333333
	ADM	1000	A01	0	1999	2	03/06/99	1	RESUMO 1	4444444444
	ADM	1000	A01	0	1999	2	06/03/99	1	RESUMO 2	4444444444
	ADM	1000	C01	0	1999	2	03/06/99	1	RESUMO 3	4444444444
	ADM	1000	C01	0	1999	2	06/03/99	1	RESUMO 3	4444444444
	FIL	3444	A01	0	1999	2	01/03/99	1	RESUMO 1	6666666666
	FIL	3444	A01	0	1999	2	03/03/99	1	RESUMO 2	6666666666
	FIS	2020	A01	0	1999	2	04/10/99	1	RESUMO 1	7777777777
	FIS	2020	A01	0	1999	2	06/03/99	1	RESUMO 2	7777777777
	FIS	2020	A01	1	1999	2	06/03/99	2	RESUMO 3	7777777777
	FIS	2020	A01	2	1999	2	06/03/99	3	RESUMO 3	7777777777
	FIS	2020	C01	0	1999	2	02/03/99	1	RESUMO 1	7777777777
	FIS	2020	C01	0	1999	2	05/03/99	1	RESUMO 2	7777777777
	FIS	2020	C01	1	1999	2	05/03/99	2	RESUMO 3	7777777777
	FIS	2020	C01	2	1999	2	05/03/99	3	RESUMO 3	7777777777
	MAT	1010	A01	0	1999	2	03/03/99	1	RESUMO 1	8888888888
	MAT	1010	A01	0	1999	2	06/03/99	1	RESUMO 2	8888888888
	MAT	1010	C01	0	1999	2	03/03/99	2	RESUMO 1	8888888888
	MAT	1010	C01	0	1999	2	06/03/99	2	RESUMO 2	8888888888

ALUNO	Matrícula	Nome	Data de Nascimento	Logradouro	Bairro	Cidade	Estado	Cep	Sexo
	1	ADNALDO LEMES	10/10/78	RUA 1	S BUENO	GOIANIA	GO	74000-000	M
	2	ADIRANA BELLINI	03/03/79	RUA 2	S CENTRAL	GOIANIA	GO	71230-000	F
	3	ALEXANDRE BARRETO	20/05/80	RUA 3	BILA NOVA	GOIANIA	GO	78000-000	M
	4	CARLOS VENAN IO	12/10/80	RUA 4	S MARISTA	GOIANIA	GO	65000-000	M
	5	DANIEL PINTO	03/03/77	RUA 5	P LUDOVICO	GOIANIA	GO	54000-000	M
	6	DANILLO LUGSTOSA	10/10/78	RUA 6	S OESTE	GOIANIA	GO	23000-500	M
	7	DENNIA ASSIS	03/03/79	RUA 7	S SUL	GOIANIA	GO	73000-000	F
	8	ELIZZIANE VIEIRA	20/05/80	RUA 8	S COIMBRA	GOIANIA	GO	74000-000	F
	9	EMERSON SALGADO	12/10/80	RUA 9	J AMERICA	GOIANIA	GO	74000-000	M
	10	FABIANO ROCHA	03/03/77	RUA 10	CAMPINAS	GOIANIA	GO	71230-000	M
	11	FABRICIO PERICLES	10/10/78	RUA 11	S BUENO	GOIANIA	GO	78000-000	M
	12	FERNANDA CURADO	03/03/79	RUA 1	S CENTRAL	GOIANIA	GO	65000-000	F
	13	FERNANDA FRANÇA	20/05/80	RUA 2	BILA NOVA	GOIANIA	GO	54000-000	F
	14	FERNANDA RIBEIRO	12/10/80	RUA 3	S MARISTA	GOIANIA	GO	23000-500	F
	15	FERNANDO ANTONIO	03/03/77	RUA 4	P LUDOVICO	GOIANIA	GO	73000-000	M
	16	GUSTAVO ALESSANDRO	10/10/78	RUA 5	S OESTE	GOIANIA	GO	74000-000	M
	17	LEANDRO HENRIQUE	03/03/79	RUA 6	S SUL	GOIANIA	GO	74000-000	M
	18	LEONARDO DIAS	20/05/80	RUA 7	S COIMBRA	GOIANIA	GO	71230-000	M
	19	LORAYNE NOVAIS	12/10/80	RUA 8	J AMERICA	GOIANIA	GO	78000-000	F
	20	LORENA CAETANO	03/03/77	RUA 9	CAMPINAS	GOIANIA	GO	65000-000	F
	21	MARCELO NARVAES	10/10/78	RUA 10	S BUENO	GOIANIA	GO	54000-000	M
	22	MARCUS BORGES	03/03/79	RUA 11	S CENTRAL	GOIANIA	GO	23000-500	M
	23	PATRICIA DE PAULA	20/05/80	RUA 1	BILA NOVA	GOIANIA	GO	73000-000	F
	24	PATRICIA SILVA	12/10/80	RUA 2	S MARISTA	GOIANIA	GO	74000-000	F
	25	RENATA CARVALHO	03/03/77	RUA 3	P LUDOVICO	GOIANIA	GO	74000-000	F
	26	RICARDO LOPES	10/10/78	RUA 4	S OESTE	GOIANIA	GO	71230-000	M
	27	RODRIGO BORGES	03/03/79	RUA 5	S SUL	GOIANIA	GO	78000-000	M
	28	STEFANE CELIAC	20/05/80	RUA 6	S COIMBRA	GOIANIA	GO	65000-000	F
	29	TACITO CLAUDIO	12/10/80	RUA 7	J AMERICA	GOIANIA	GO	54000-000	M
	30	TATIANA PIRES	03/03/77	RUA 8	CAMPINAS	GOIANIA	GO	23000-500	F
	31	WALKER DA SILVA	15/02/80	RUA 9	DERGO	GOIANIA	GO	73000-000	M

MATRICULA	Codigo Curso	Codigo Disciplina	Numero Turma	Numero SubTurma	Matricula	Data Matricula	Nota Final
	CMP	4070	A01	0		20/06/98	
	9999999						

DISCIPLINA	Codigo Curso	Codigo	Nome	Creditos	Prelecao	Laboratorio
	CMP	4144	Banco de Dados I	04	02	02
	CMP	4153	Analise e Projeto de Sistemas	06	00	00
	CMP	4152	Banco de Dados II	06	04	02
	CMP	4111	Int à Ciencia da Computação I	06	04	02
	CMP	4120	Telep e Rede de Computadores	06	04	02
	CMP	4121	Int à Ciência da Computação II	06	04	02
	CMP	4131	Estrutura de Dados I	06	04	02
	CMP	4133	Sistemas Digitais I	06	04	02
	CMP	4134	Paradigmas de Programação	06	04	02
	CMP	4142	Linguagens de Montagem	06	04	02
	CMP	4143	Estrutura de Dados II	06	04	02
	CMP	4151	Sistemas Operacionais I	06	04	02
	CMP	4154	Compiladores	06	04	02
	CMP	4155	Complexidade de Algoritmos	06	04	02
	CMP	4157	Pesquisa Operacional I	04	02	02

ALUNO	Matricula	Nome	Data de Nascimento	Sexo	Logradouro	Bairro	Cidade	Estado	Cep	Fone
	111111111	Ana Carolina Rezende	10/10/80	F	Rua 18 N.430	Centro	Goiania	GO	74000-001	2513456
	222222222	Ana Claudia Pimenta	03/07/79	F	Rua 120 N.670	Centro	Goiania	GO	74000-002	
	333333333	Ana Maria dos Anjos	08/01/80	F	Rua C-340	Jardim America	Goiania	GO	74000-003	9876769
	444444444	Ana Lucia de Araujo	15/02/79	F	Rua C-460	Jardim America	Goiania	GO	74000-004	9876578
	555555555	Ana Vieira Peixoto	12/03/78	F	Rua C-320	Nova Suíça	Goiania	GO	74000-005	8765434
	666666666	Ana Paula Mamedes	25/03/78	F	Rua T-49	Setor Bueno	Goiania	GO	74000-006	2345678
	777777777	Berenice Figueiredo	23/09/75	F	Rua T-43	Setor Bueno	Goiania	GO	74000-007	8769099
	888888888	Beatriz Ranz	23/09/87	F	Rua T-24	Setor Bueno	Goiania	GO	74000-008	1234567
	999999999	Carla Gomes Fonseca	12/12/80	F	Rua T-24	Setor Bueno	Goiania	GO	74000-009	2345678
	101010101	Celia Bezerra Rezende	12/11/81	F	Rua T-25 N.43	Setor Bueno	Goiania	GO	74000-010	3456789
	202020202	Celina Mesquita Freitas	11/11/78	F	Rua T-25 N.98	Setor Bueno	Goiania	GO	74000-011	5678901
	303030303	Celimene Mota Gondim	13/12/78	F	Rua T-25 N.100	Setor Bueno	Goiania	GO	74000-012	6789012
	404040404	Carmem Vistosa Moreira	02/02/77	F	Rua T-40 N.107	Setor Bueno	Goiania	GO	74000-013	7890123
	505050505	Carmem Lucia Correa	03/03/80	F	Rua T-37 N.203	Setor Bueno	Goiania	GO	74000-014	8901234
	606060606	Carmem Rosa dos Anjos	04/04/80	F	Rua T-78 N.450	Setor Buendo	Goiania	GO	74000-015	9012345
	707070707	Danila Freitas Mascarenha	05/05/78	F	Rua 43	Centro	Goiania	GO	74000-016	8098765
	808080808	Dalva Moreira Machado	06/06/78	F	Rua 70 N.43	Centro	Goiania	GO	74000-017	9090900
	909090909	Diva Maria dos Santos	07/07/77	F	Rua 80 N. 456	Centro	Goiania	GO	74000-018	1019999
	000111111	Dalva Carolina Simões	08/08/77	F	Rua 90 N.780	Centro	Goiania	GO	74000-019	2019999
	000222222	Denise Machado Rodrigues	09/03/78	F	Rua 120 N. 800	Centro	Goiania	GO	74000-020	2029999
	000333333	Dolarice Rodrigues Cunha	10/10/78	F	Rua 700 N. 440	Centro	Goiania	GO	74000-021	2039999
	000444444	Delia Cunha Rezende	09/08/78	F	Rua 130 N. 430	Centro	Goiania	GO	74000-022	2049999
	000555555	Dirce dos Amjos	10/08/76	F	Rua 230 N.480	Setor Coimbra	Goiania	GO	74000-023	2059999
	000666666	Dulce Salomão Rezende	23/09/77	F	Rua 237 N.140	Setor Coimbra	Goiania	GO	74000-024	2069999
	000777777	Dolores Cravo Reis	23/10/78	F	Rua 240 N.170	Setor Coimbra	Goiânia	GO	74000-025	2099999
	000888888	Divane Marques Pereira	24/12/78	F	Rua 137 N. 767	Setor Coimbra	Goiania	GO	74000-026	2100000
	000999999	Denia Garcia Cardoso	17/03/79	F	Rua 435 N.	Setor	Goiania	GO	74000-027	2100001

					870	Coimbra				
	000011111	Dilma Guimaraes Teixeira	12/12/81	F	Rua 799 N. 830	Vila Nova	Goiania	GO	74000-028	2100002
	000022222	Dagmar Ferreira dos Santos	01/01/81	F	Rua 230 N. 444	Vila Nova	Goiania	GO	74000-029	2100003
	000033333	Dora Fernandes Ribeiro	01/04/81	F	Rua 122 N. 333	Vila Nova	Goiania	GO	74000-030	2100004
	000044444	Eunice Carmem Mendes	03/02/81	F	Rua 333 N.444	Vila Nova	Goiania	GO	74000-031	2100005
	000055555	Erika Mendes Sobrinho	03/03/81	F	Rua 111 N.111	Vila Nova	Goiania	GO	74000-032	2100006
	000066666	Elma Mahcado Goes	02/01/80	F	Rua 222 N.222	Vila Nova	Goiania	GO	74000-033	2100007
	000077777	Edna Rodrigues Machado	03/03/81	F	Rua 333 N.333	Vila Nova	Goiania	GO	74000-034	2100008
	000088888	Eugenica Franga Candido	03/01/78	F	Rua 555 N.555	Vila Nova	Goiania	GO	74000-035	2100009
	000099999	Evandssa Magalhaes Rosa	17/08/77	F	Rua P-12 N.12	Setor Funcionarios	Goiania	GO	74000-036	2100010
	000001111	Elvira Garrotte Mascarenhas	12/12/76	F	Rua P-13 N.13	Setor Funcionarios	Goiania	GO	74000-037	2100011
	000002222	Francisca Rodrigues Cunha	21/06/69	F	Rua P-14 N.14	Setor Funcionarios	Goiania	GO	74000-038	2100012
	000003333	Fernanda Pacheco Costa	22/06/70	F	Rua P-15 N.15	Setor Funcionarios	Goiania	GO	74000-039	2100013
	000004444	Flavia Cascudo da Costa	12/04/76	F	Rua P-16 N.16	Setor Funcionarios	Goiania	GO	74000-044	2100014
	000005555	Fatima Cascão Vieira	04/08/76	F	Rua P-43 N.70	Setor Funcionarios	Goiania	GO	74000-041	2100015
	000006666	Fabíola dos Reis Lins	05/05/73	F	Rua P_45 N.32	Setor Funcionarios	Goiania	GO	74000-042	2100016
	000007777	Filomena Cardoso Amaral	15/07/74	F	Rua P-56 N.43	Setor Funcionarios	Goiania	GO	74000-043	2100017
	000008888	Fabiana Cristovam Pimenta	12/03/78	F	Rua P_12 N. 120	Setor Funcionarios	Goiania	GO	74000-044	2100018
	000009999	Fabíola Rezende Cardoso	13/03/77	F	Rua P_13 N.340	Setor Funcionarios	Goiania	GO	74000-045	2100019
	000000111	Gisele Mendonça Ribeiro	03/04/75	F	Rua T-35 N.777	Setor Bueno	Goiania	GO	74000-046	2100020
	000000222	Gleice Quincas Batista	23/07/80	F	Rua T-40 N.44	Setor Bueno	Goiania	GO	74000-047	2100030
	000000333	Gilda Pereira Guimaraes	13/07/75	F	Rua T-55 N.55	Setor Bueno	Goiania	GO	74000-048	2100031
	000000444	Gina Guimaraes Rosa	12/06/68	F	Rua T-55 N.40	Setor Bueno	Goiania	GO	74000-049	2100032
	000000555	Gilka Ferreira Mendes	30/05/74	F	Rua T-40 N.41	Setor Bueno	Goiania	GO	74000-050	2100033
	000000666	Geralda Zacharias	12/09/69	F	Rua T-33 N.42	Setor Bueno	Goiania	GO	74000-051	2100034
	000000777	Galba Mendes Ribas	13/04/74	F	Rua T-45 N.78	Setor Bueno	Goiania	GO	74000-052	2100035
	000000888	Gilmara Rezende Ribeiro	12/04/80	F	Rua T-40 N.78	Setor Bueno	Goiania	GO	74000-053	2100036

	000000999	Gersina Rosa dos Anjos	30/03/74	F	Rua T-41 N.40	Setor Bueno	Goiania	GO	74000-054	2100037
	000000011	Helena de Freitas Lima	31/03/76	F	Rua T-41 N.41	Setor Bueno	Goiania	GO	74000-055	2100038
	000000022	Hozana Maria Pimentel	22/04/78	F	Rua T-45 N.43	Setor Bueno	Goiania	GO	74000-056	2100039
	000000033	Helia Fernandes Martins	30/04/76	F	Rua T-01 N.44	Setor Bueno	Goiania	GO	74000-057	2100040
	000000044	Hermínia Fragoso	25/10/76	F	Rua T-02 N.55	Setor Bueno	Goiania	GO	74000-058	2100041
	000000055	Hilda Francisco Masarda	30/10/76	F	Rua T-03 N.66	Setor Bueno	Goiania	GO	74000-059	2100042
	000000066	Hortencia Flubius	29/08/74	F	Rua T-04 N.77	Setor Bueno	Goiania	GO	74000-060	2100043
	000000077	Hilma Magalhaes	25/07/79	F	Rua T-05 N.88	Setor Bueno	Goiania	GO	74000-061	2100044
	000000088	Helena Guimaraes Peixoto	01/01/70	F	Rua T-06 N.99	Setor Bueno	Goiania	GO	74000-062	2100045
	000000099	Helia Ribiero Cristina	02/03/76	F	Rua T-07 N.100	Setor Bueno	Goiania	GO	74000-063	2100046
	000000001	Ivana da Conceição	03/06/74	F	Rua 1 N.2	Centro	Goiania	GO	74000-064	2100047
	000000002	Ivete Rodrigues Cunha	23/04/72	F	Rua 2 N.40	Centro	Goiania	GO	74000-065	2100048
	000000003	Izis Marins Gomes	24/05/78	F	Rua 3 N.3	Centro	Goiania	GO	74000-066	2100049
	000000004	Ivone Rosa dos Anos	25/04/78	F	Rua 4 N.4	Centro	Goiania	GO	74000-067	2100050
	000000005	Ivonilde Maria da Costa	04/07/65	F	Rua 5 N.5	Centro	Goiania	GO	74000-068	2100051
	000000006	Iris Gouveia Jatoba	31/03/74	F	Rua 6 N.6	Centro	Goiania	GO	74000-069	2100052
	000000007	Irany Pimenta Cedro	21/04/74	F	Rua 7 N.7	Centro	Goiania	GO	74000-070	2100053
	000000008	Irene Rosa Pinheiro	05/05/73	F	Rua 8 N.8	Centro	Goiania	GO	74000-071	2100054
	000000009	Isolina Marques	27/05/78	F	Rua 9 N.9	Centro	Goiania	GO	74000-072	2100055
	011111111	Janaina Melissa Reis	21/05/78	F	Rua 10 N.10	Centro	Goiania	GO	74000-073	2100056
	022222222	Joana Marques Ferreira	03/09/79	F	Rua 11 N.11	Centro	Goiania	GO	74000-074	2100057
	033333333	Jordana Pires Ribeiro	04/09/77	F	Rua 12 N.12	Centro	Goiania	GO	74000-075	2100058
	044444444	Joselice Pinheiro Machado	09/05/73	F	Rua 13 N.13	Centro	Goiania	GO	74000-076	2100059
	055555555	Judite Maria Medeiros	12/12/81	F	Rua 14 N.14	Centro	Goiania	GO	74000-077	2100060
	066666666	Josefa Ferreira Ros	15/12/69	F	Rua 15 N.15	Centro	Goiania	GO	74000-078	2100061
	077777777	Jurema Madalena Viera	07/12/69	F	Rua 16 N.16	Centro	Goiania	GO	74000-079	2100062
	088888888	Julia Maria Madalena	08/11/72	F	Rua 17 N.17	Centro	Goiania	GO	74000-080	2100063
	099999999	Juliete Fernandes Castro	12/01/69	F	Rua 18 N.18	Centro	Goiania	GO	74000-081	2100064
	010111111	Ludimila Carmen Reis	27/10/68	F	Rua 19 N.19	Centro	Goiania	GO	74000-082	2100065
	010222222	Lorena Freitas Borges	23/06/72	F	Rua 20 N.20	Centro	Goiania	GO	74000-083	2100066
	010333333	Letícia dos Santos	13/04/64	F	Rua 21 N.21	Centro	Goiania	GO	74000-084	2100067
	010444444	Luiza Ribeiro de Freitas	23/05/81	F	Rua 22 N.22	Centro	Goiania	GO	74000-085	2100068
	010555555	Luzia Fernandes da Luz	21/03/64	F	Rua 23 N.23	Centro	Goiania	GO	74000-086	2100069
	010666666	Lara dos Santos Pinho	22/04/65	F	Rua 24 N.24	Centro	Goiania	GO	74000-087	2100070
	010777777	Livia Martins Lima	12/09/74	F	Rua 25 N.25	Centro	Goiania	GO	74000-087	2100071
	010888888	Leila Lorenço Marques	25/10/75	F	Rua 26 N.26	Centro	Goiania	GO	74000-088	2100072
	010999999	Liria da Luz Mendes	22/12/73	F	Rua 27 N.27	Centro	Goiania	GO	74000-089	2100073
	010011111	Maria Ines Marques	23/03/79	F	Rua 28 N.28	Centro	Goiania	GO	74000-090	2100074

banco de Dados I

127

	010022222	Maria Matildes Ramos	17/09/74	F	Rua 29 N.29	Centro	Goiania	GO	74000-091	2100075
	010033333	Maria das Dores Rocha	12/05/74	F	Rua 30 N.30	Centro	Goiania	GO	74000-092	2100076
	010044444	Maria Madalena Pessego	16/02/79	F	Rua 31 N.31	Centro	Goiania	GO	74000-093	2100077
	010055555	Maria de Fatima Rosa	15/04/72	F	Rua 32 N.32	Centro	Goiania	GO	74000-094	2100078
	010066666	Maria Odete Fernandes	04/09/62	F	Rua 33 N.33	Centro	Goiania	GO	74000-095	2100079
	010077777	Maria Claudia Bernardes	08/05/71	F	Rua 34 N.34	Centro	Goiania	GO	74000-096	2100080
	010088888	Maria Mercedes Marta	03/09/67	F	Rua 35 N.35	Centro	Goiania	GO	74000-097	2100081
	010099999	Madalena Cintra	07/04/78	F	Rua 36 N.36	Centro	Goiania	GO	74000-098	2100082
	010001111	Patricia Pires do Rio	10/10/79	F	Rua 37 N.37	Centro	Goiania	GO	74000-099	2100083
	010002222	Paula Cândido Ferreira	11/11/74	F	Rua 38 N.38	Centro	Goiania	GO	74000-100	2100084



TURMA	Ano Semestre	Numero Semestre	Codigo Curso	Codigo Disciplina	Numero	Numero Sub Turma
	1998	1	CMP	4144	A01	1
	1998	1	CMP	4144	A01	2
	1998	1	CMP	4144	A01	3
	1998	1	CMP	4144	C01	1
	1998	1	CMP	4144	C01	2
	1998	1	CMP	4144	C02	1
	1998	1	CMP	4144	C02	2
	1998	2	CMP	4144	A01	1
	1998	2	CMP	4144	A01	2
	1998	2	CMP	4144	C01	1
	1998	2	CMP	4144	C01	2
	1998	2	CMP	4144	C02	1
	1998	2	CMP	4144	C02	2
	1999	1	CMP	4144	A01	1
	1999	1	CMP	4144	A01	2
	1999	1	CMP	4144	C01	1
	1999	1	CMP	4144	C01	2
	1999	1	CMP	4144	C02	1
	1999	1	CMP	4144	C02	2
	1999	2	CMP	4144	A01	1
	1999	2	CMP	4144	A01	2
	1999	2	CMP	4144	C01	1
	1999	2	CMP	4144	C01	2
	1999	2	CMP	4144	C02	1
	1999	2	CMP	4144	C02	2
	1998	1	CMP	4153	A01	
	1998	1	CMP	4153	C01	
	1998	2	CMP	4153	A01	
	1998	2	CMP	4153	C01	
	1999	1	CMP	4153	A01	
	1999	1	CMP	4153	C01	
	1999	2	CMP	4153	A01	
	1999	2	CMP	4153	C01	
	1998	1	CMP	4152	A01	1
	1998	1	CMP	4152	A01	2
	1998	1	CMP	4152	C01	1
	1998	1	CMP	4152	C01	2
	1998	1	CMP	4152	C02	1
	1998	1	CMP	4152	C02	2
	1998	2	CMP	4152	A01	1

## banco de Dados I

1998	2	CMP	4152	A01	2
1998	2	CMP	4152	C01	1
1998	2	CMP	4152	C01	2
1998	2	CMP	4152	C02	1
1998	2	CMP	4152	C02	2
1999	1	CMP	4152	A01	1
1999	1	CMP	4152	A01	2
1999	1	CMP	4152	C01	1
1999	1	CMP	4152	C01	2
1999	1	CMP	4152	C02	1
1999	1	CMP	4152	C02	2
1999	2	CMP	4152	A01	1
1999	2	CMP	4152	A01	2
1999	2	CMP	4152	C01	1
1999	2	CMP	4152	C01	2
1999	2	CMP	4152	C02	1
1999	2	CMP	4152	C02	2
1998	1	CMP	4111	A01	1
1998	1	CMP	4111	A01	2

MATRICULA	Matricula Aluno	Ano Semestre	Numero Semestre	Codigo Curso	Codigo Disciplina	Numero Turma	Sub Turma	DataDa Matricula	N1	N2
	111111111	1998	1	CMP	4144	A01	1	20/01/98	6,5	4,2
	222222222	1998	1	CMP	4144	A01	1	20/01/98	2,0	5,0
	333333333	1998	1	CMP	4144	A01	1	21/01/98	7,0	6,9
	444444444	1998	1	CMP	4144	A01	1	21/01/98	5,0	5,5
	555555555	1998	1	CMP	4144	A01	1	22/01/98	6,0	6,0
	666666666	1998	1	CMP	4144	A01	1	23/01/98	7,0	8,0
	777777777	1998	1	CMP	4144	A01	2	15/01/98		
	888888888	1998	1	CMP	4144	A01	2	15/01/98		
	999999999	1998	1	CMP	4144	A01	2	16/01/99		
	101010101	1998	1	CMP	4144	C01	1	12/01/98		
	202020202	1998	1	CMP	4144	C01	1	12/01/98		
	303030303	1998	1	CMP	4144	C01	1	13/01/98		
	404040404	1998	1	CMP	4144	C01	1	14/01/98		
	505050505	1998	1	CMP	4144	C01	1	15/01/98		
	606060606	1998	1	CMP	4144	C01	2	20/01/98		
	707070707	1998	1	CMP	4144	C01	2	20/01/98		
	808080808	1998	1	CMP	4144	C01	2	21/01/98		
	909090909	1998	1	CMP	4144	C01	2	21/01/98		
	000111111	1998	1	CMP	4144	C02	1	22/01/98		
	000222222	1998	1	CMP	4144	C02	1	23/01/98		
	000333333	1998	1	CMP	4144	C02	1	15/01/98		
	000444444	1998	1	CMP	4144	C02	1	15/01/98		

## banco de Dados I

	000555555	1998	1	CMP	4144	C02	1	16/01/99		
	000666666	1998	1	CMP	4144	C02	2	12/01/98		
	000777777	1998	1	CMP	4144	C02	2	12/01/98		
	000888888	1998	1	CMP	4144	C02	2	13/01/98		
	000999999	1998	1	CMP	4144	C02	2	14/01/98		
	000011111	1998	2	CMP	4144	A01	1	15/07/98		
	000022222	1998	2	CMP	4144	A01	1	15/07/98		
	000033333	1998	2	CMP	4144	A01	1	15/07/98		
	000044444	1998	2	CMP	4144	A01	1	16/07/98		
	000055555	1998	2	CMP	4144	A01	1	17/07/98		
	000066666	1998	2	CMP	4144	A01	2	15/07/98		
	000077777	1998	2	CMP	4144	A01	2	15/07/98		
	000088888	1998	2	CMP	4144	A01	2	15/07/98		
	000099999	1998	2	CMP	4144	A01	2	16/07/98		
	000001111	1998	2	CMP	4144	C01	1	17/07/98		
	000002222	1998	2	CMP	4144	C01	1	15/07/98		
	000003333	1998	2	CMP	4144	C01	1	15/07/98		
	000004444	1998	2	CMP	4144	C01	1	15/07/98		
	000005555	1998	2	CMP	4144	C01	1	16/07/98		
	000006666	1998	2	CMP	4144	C01	2	17/07/98		
	000007777	1998	2	CMP	4144	C01	2	15/07/98		
	000008888	1998	2	CMP	4144	C01	2	15/07/98		
	000009999	1998	2	CMP	4144	C01	2	15/07/98		
	000000111	1998	2	CMP	4144	C02	1	16/07/98		
	000000222	1998	2	CMP	4144	C02	1	17/07/98		
	000000333	1998	2	CMP	4144	C02	1	15/07/98		
	000000444	1998	2	CMP	4144	C02	1	15/07/98		
	000000555	1998	2	CMP	4144	C02	1	15/07/98		
	000000666	1998	2	CMP	4144	C02	2	16/07/98		
	000000777	1998	2	CMP	4144	C02	2	17/07/98		
	000000888	1998	2	CMP	4144	C02	2	15/07/98		
	000000999	1998	2	CMP	4144	C02	2	15/07/98		
	111111111	1998	2	CMP	4152	A01	1	15/07/98		
	222222222	1998	2	CMP	4152	A01	1	16/07/98		
	333333333	1998	2	CMP	4152	A01	1	17/07/98		
	444444444	1998	2	CMP	4152	A01	1	15/07/98		
	555555555	1998	2	CMP	4152	A01	1	15/07/98		
	666666666	1998	2	CMP	4152	A01	1	15/07/98		
	777777777	1998	2	CMP	4152	A01	2	16/07/98		
	888888888	1998	2	CMP	4152	A01	2	17/07/98		
	101010101	1998	2	CMP	4152	C01	1	15/07/98		
	202020202	1998	2	CMP	4152	C01	1	15/07/98		
	303030303	1998	2	CMP	4152	C01	1	15/07/98		

	404040404	1998	2	CMP	4152	C01	1	16/07/98		
	505050505	1998	2	CMP	4152	C01	1	17/07/98		
	606060606	1998	2	CMP	4152	C01	2	15/07/98		
	707070707	1998	2	CMP	4152	C01	2	15/07/98		
	808080808	1998	2	CMP	4152	C01	2	15/07/98		
	909090909	1998	2	CMP	4152	C01	2	16/07/98		
	000111111	1998	2	CMP	4152	C02	1	17/07/98		
	000222222	1998	2	CMP	4152	C02	1	15/07/98		
	000333333	1998	2	CMP	4152	C02	1	15/07/98		
	000444444	1998	2	CMP	4152	C02	1	15/07/98		
	000555555	1998	2	CMP	4152	C02	1	16/07/98		
	000666666	1998	2	CMP	4152	C02	2	17/07/98		
	000777777	1998	2	CMP	4152	C02	2	15/07/98		
	000888888	1998	2	CMP	4152	C02	2	15/07/98		
	000999999	1998	2	CMP	4152	C02	2	15/07/98		
	111111111	1999	1	CMP	4153	A01		16/01/99		
	222222222	1999	1	CMP	4153	A01		17/01/99		
	333333333	1999	1	CMP	4153	A01		16/01/99		
	444444444	1999	1	CMP	4153	A01		17/01/99		
	555555555	1999	1	CMP	4153	A01		16/01/99		
	666666666	1999	1	CMP	4153	A01		17/01/99		
	777777777	1999	1	CMP	4153	A01		16/01/99		
	888888888	1999	1	CMP	4153	A01		17/01/99		
	101010101	1999	2	CMP	4153	C01		16/07/99		
	202020202	1999	2	CMP	4153	C01		17/07/99		
	303030303	1999	2	CMP	4153	C01		16/07/99		
	404040404	1999	2	CMP	4153	C01		17/07/99		
	505050505	1999	2	CMP	4153	C01		16/07/99		
	606060606	1999	2	CMP	4153	C01		17/07/99		
	707070707	1999	2	CMP	4153	C01		16/07/99		
	808080808	1999	2	CMP	4153	C01		17/07/99		
	909090909	1999	2	CMP	4153	C01		16/07/99		
	000111111	1999	2	CMP	4153	C02		17/07/99		
	000222222	1999	2	CMP	4153	C02		16/07/99		
	000333333	1999	2	CMP	4153	C02		17/07/99		
	000444444	1999	2	CMP	4153	C02		16/07/99		
	000555555	1999	2	CMP	4153	C02		17/07/99		
	000666666	1999	2	CMP	4153	C02		16/07/99		
	000777777	1999	2	CMP	4153	C02		17/07/99		
	000888888	1999	2	CMP	4153	C02		16/07/99		
	000999999	1999	2	CMP	4153	C02		17/07/99		

banco de Dados I

AULA	Ano Semestre	Numero Semestre	Codigo Curso	Codigo Disciplina	CodigoTur ma	Sub Turma	Data	Nume ro	Cpf Professor	ResumoAula
	1998	1	CMP	4144	A01	1	03/03/99	1		
	1998	1	CMP	4144	A01	1	05/03/99	1		
	1998	1	CMP	4144	A01	1	10/03/99	1		
	1998	1	CMP	4144	A01	1	12/03/99	1		
	1998	1	CMP	4144	A01	1	17/03/99	1		
	1998	1	CMP	4144	A01	1	19/03/99	1		
	1998	1	CMP	4144	A01	2	05/07/99	1		
	1998	1	CMP	4144	A01	2	12/05/99	1		
	1998	1	CMP	4144	A01	2	19/05/99	1		
	1998	1	CMP	4144	C01	1	01/03/99	1		
	1998	1	CMP	4144	C01	1	03/03/99	1		
	1998	1	CMP	4144	C01	1	08/03/99	1		
	1998	1	CMP	4144	C01	1	10/03/99	1		
	1998	1	CMP	4144	C01	1	15/03/99	1		
	1998	1	CMP	4144	C01	2	03/03/99	1		
	1998	1	CMP	4144	C01	2	10/03/99	1		
	1998	1	CMP	4144	C01	2	17/03/99	1		
	1998	1	CMP	4144	C01	2	19/03/99	1		
	1998	1	CMP	4144	C02	1	02/04/99	1		
	1998	1	CMP	4144	C02	1	05/04/99	1		
	1998	1	CMP	4144	C02	1	08/04/99	1		
	1998	1	CMP	4144	C02	1	11/04/99	1		
	1998	1	CMP	4144	C02	1	14/04/99	1		
	1998	1	CMP	4144	C02	2	17/04/99	1		
	1998	1	CMP	4144	C02	2	20/04/99	1		
	1998	1	CMP	4144	C02	2	23/04/99	1		
	1998	1	CMP	4144	C02	2	26/04/99	1		
	1998	2	CMP	4144	A01	1	01/08/99	1		
	1998	2	CMP	4144	A01	1	03/08/99	1		
	1998	2	CMP	4144	A01	1	05/08/99	1		
	1998	2	CMP	4144	A01	1	07/08/99	1		
	1998	2	CMP	4144	A01	1	09/08/99	1		
	1998	2	CMP	4144	A01	2	03/08/99	1		
	1998	2	CMP	4144	A01	2	05/08/99	1		
	1998	2	CMP	4144	A01	2	07/08/99	1		
	1998	2	CMP	4144	A01	2	09/08/99	1		
	1998	2	CMP	4144	C01	1	01/08/99	1		
	1998	2	CMP	4144	C01	1	03/08/99	1		
	1998	2	CMP	4144	C01	1	05/08/99	1		
	1998	2	CMP	4144	C01	1	07/08/99	1		
	1998	2	CMP	4144	C01	1	09/08/99	1		
	1998	2	CMP	4144	C01	2	03/08/99	1		
	1998	2	CMP	4144	C01	2	05/08/99	1		

## banco de Dados I

	1998	2	CMP	4144	C01	2	07/08/99	1		
	1998	2	CMP	4144	C01	2	09/08/99	1		
	1998	2	CMP	4144	C02	1	01/08/99	1		
	1998	2	CMP	4144	C02	1	03/08/99	1		
	1998	2	CMP	4144	C02	1	05/08/99	1		
	1998	2	CMP	4144	C02	1	07/08/99	1		
	1998	2	CMP	4144	C02	1	09/08/99	1		
	1998	2	CMP	4144	C02	2	03/08/99	1		
	1998	2	CMP	4144	C02	2	05/08/99	1		
	1998	2	CMP	4144	C02	2	07/08/99	1		
	1998	2	CMP	4144	C02	2	09/08/99	1		
	1998	2	CMP	4152	A01	1	01/08/99	1		
	1998	2	CMP	4152	A01	1	03/08/99	1		
	1998	2	CMP	4152	A01	1	05/08/99	1		
	1998	2	CMP	4152	A01	1	07/08/99	1		
	1998	2	CMP	4152	A01	1	09/08/99	1		
	1998	2	CMP	4152	A01	1	03/08/99	1		
	1998	2	CMP	4152	A01	2	05/08/99	1		
	1998	2	CMP	4152	A01	2	07/08/99	1		
	1998	2	CMP	4152	C01	1	06/08/99	1		
	1998	2	CMP	4152	C01	1	01/08/99	1		
	1998	2	CMP	4152	C01	1	03/08/99	1		
	1998	2	CMP	4152	C01	1	05/08/99	1		
	1998	2	CMP	4152	C01	1	07/08/99	1		
	1998	2	CMP	4152	C01	2	09/08/99	1		
	1998	2	CMP	4152	C01	2	03/08/99	1		
	1998	2	CMP	4152	C01	2	05/08/99	1		
	1998	2	CMP	4152	C01	2	07/08/99	1		
	1998	2	CMP	4152	C02	1	09/08/99	1		
	1998	2	CMP	4152	C02	1	01/08/99	1		
	1998	2	CMP	4152	C02	1	03/08/99	1		
	1998	2	CMP	4152	C02	1	05/08/99	1		
	1998	2	CMP	4152	C02	1	07/08/99	1		
	1998	2	CMP	4152	C02	2	10/08/99	1		
	1998	2	CMP	4152	C02	2	03/08/99	1		
	1998	2	CMP	4152	C02	2	05/08/99	1		
	1998	2	CMP	4152	C02	2	07/08/99	1		
	1999	1	CMP	4153	A01		05/01/99	1		
	1999	1	CMP	4153	A01		05/01/99	2		
	1999	1	CMP	4153	A01		06/01/99	1		
	1999	1	CMP	4153	A01		06/01/99	2		
	1999	1	CMP	4153	A01		07/01/99	1		
	1999	1	CMP	4153	A01		07/01/99	2		
	1999	1	CMP	4153	A01		08/01/99	1		
	1999	1	CMP	4153	A01		09/01/99	1		

## banco de Dados I

	1999	2	CMP	4153	C01		05/01/99	1		
	1999	2	CMP	4153	C01		05/01/99	2		
	1999	2	CMP	4153	C01		06/01/99	1		
	1999	2	CMP	4153	C01		06/01/99	2		
	1999	2	CMP	4153	C01		07/01/99	1		
	1999	2	CMP	4153	C01		07/01/99	2		
	1999	2	CMP	4153	C01		08/01/99	1		
	1999	2	CMP	4153	C01		09/01/99	1		
	1999	2	CMP	4153	C01		09/01/99	2		
	1999	2	CMP	4153	C02		05/01/99	1		
	1999	2	CMP	4153	C02		05/01/99	2		
	1999	2	CMP	4153	C02		06/01/99	1		
	1999	2	CMP	4153	C02		06/01/99	2		
	1999	2	CMP	4153	C02		07/01/99	1		
	1999	2	CMP	4153	C02		07/01/99	2		
	1999	2	CMP	4153	C02		08/01/99	1		
	1999	2	CMP	4153	C02		09/01/99	1		
	1999	2	CMP	4153	C02		09/01/99	2		

FREQUE NCIA	Ano Semestre	Numero Semestre	Codigo Curso	Codigo Disciplina	Codigo Turma	Numero Sub Turma	Data Aula	Numero Aula	Matricula Aluno	Ind Presença
	1998	1	CMP	4144	A01	1	03/03/99	1	111111111	
									222222222	
									333333333	
									555555555	
									666666666	
	1998	1	CMP	4144	A01	1	05/03/99	1	111111111	
									333333333	
									444444444	
									555555555	
									666666666	
	1998	1	CMP	4144	A01	1	10/03/99	1	111111111	
									333333333	
									444444444	
									555555555	
									666666666	
	1998	1	CMP	4144	A01	1	12/03/99	1	111111111	
									333333333	
									444444444	
									555555555	
									666666666	
	1998	1	CMP	4144	A01	1	17/03/99	1	111111111	

banco de Dados I

135

									33333333	
									44444444	
									55555555	
									66666666	
	1998	1	CMP	4144	A01	1	19/03/99	1	11111111	
									33333333	
									44444444	
									55555555	
									66666666	





# EXERCÍCIOS SQL



## Banco de Dados – ACADÊMICO – Parte I

Tendo como referência as definições do banco de dados **ACADEMICO** responda as questões apresentadas a seguir, usando as operações da Álgebra Relacional e comandos SQL.

1. Escreva uma operação de inserção na relação **Professor** que viola a restrição de integridade de entidade.
2. Escreva uma operação de inserção na relação **Professor** que viola restrições de integridade referencial.
3. Escreva uma operação de inserção na relação **Aluno** que viola restrições de integridade de chave.
4. Escreva uma operação de exclusão na relação **Departamento** que viola restrições de integridade referencial.
5. Escreva uma operação de alteração na relação **Aluno** que viola restrições de domínio.
6. Obtenha uma relação com todos os atributos dos professores do departamento CMP.
7. Obtenha uma relação com Cpf, Nome e Endereço dos professores do departamento ADMINISTRAÇÃO.
8. Obtenha uma relação das disciplinas dos cursos de CIÊNCIA DA COMPUTAÇÃO e FÍSICA.
9. Obtenha a relação DEP com os atributos Sigla e Nome do Departamento que devem corresponder respectivamente aos atributos Código e Nome da relação DEPARTAMENTO.
10. Dado as duas relações abaixo, mostre o conteúdo da relação RESULTADO obtida da operação  $DEP1 \cup DEP2$ .

DEP1	Codigo	Nome
	CMP	CIÊNCIA DA COMPUTAÇÃO
	FIT	FILOSOFIA E TEOLOGIA
	ADM	ADMINISTRAÇÃO
	ENG	ENGENHARIA

DEP2	Codigo	Nome
	ADM	ADMINISTRAÇÃO
	ENG	ENGENHARIA
	ECO	ECONOMIA

[illegible]

11. Dados as duas relações a seguir, mostre o conteúdo da relação RESULTADO obtida da operação  $A \cap B$

A	Matricula	Nome
	1	FERREIRA NETO
	2	FRANCISCO RIBEIRO
	3	CARLOS GOMES
	4	REGINA DUARTE
	8	BONAPARTE FIGUEIREDO
	5	CARLOS VEIGA

B	Matricula	Nome
	1	FERREIRA NETO
	7	FRANCISCO RIBEIRO
	4	CARLOS GOMES
	2	REGINA DUARTE
	8	BONAPARTE FIGUEIREDO
	5	CARLOS VEIGA

RESULTADO	Matricula	Nome

12. Dado as duas relações abaixo mostre a relação RESULTADO obtida da operação A – B

A	Matricula
	1
	2
	3
	4
	5
	6
	7

B	Matricula
	1
	3
	2
	4

RESULTADO	Matricula

13. Dado as relações A e B, mostrar o conteúdo da relação RESULTADO após a sequência de operações a seguir:

RESULTADO ← (A ∩ B) - B

A	Matricula
	1
	2
	3
	4
	5
	6
	7

B	Matricula
	1
	5
	3
	10
	11
	15
	9

14. Dado as relações A e B abaixo, mostrar o conteúdo da relação C após as a sequência de operações a seguir:

C ← (A ∪ B) ∩ B

A	Matricula
	1
	2
	3
	4
	5
	6
	7

B	Matricula
	1
	3
	2
	4

C	Matricula

15. Mostre a relação RESULTADO resultante do produto da sequencia de operações a seguir:

WORK1 ← SELECT<sub>(Codigo = "CMP")</sub> (CURSO)

WORK2 ← WORK1 X DISCIPLINA

16. Mostrar o conteúdo da relação resultante da operação a seguir:

CURSO JOIN<sub>(Codigo = CodigoCurso)</sub> DISCIPLINA

17. Obter uma relação com os alunos matriculados na disciplina Engenharia de Software do curso CMP. A relação resultante deve conter: Matricula do aluno, Nome do aluno, Codigo e Nome da Disciplina e o Codigo do Curso.
18. Obter uma relação com o número de faltas do aluno em cada disciplina que ele está matriculado: A relação resultante deve conter: Matricula do Aluno, Data da Aula, Codigo e Nome da Disciplina e Codigo do Curso.
19. Obter uma relação com a quantidade dos alunos matriculados na disciplina Banco de Dados I do curso CMP.
20. Quais disciplinas não têm nenhum aluno matriculado.
21. Quais são as disciplinas lecionadas pelo professor Ivon Canedo. Mostrar todos os atributos das disciplinas.
22. Quais alunos foram reprovados em cada disciplina cursada (Nota Final <5,0). Mostrar: Matricula e Nome do Aluno, Codigo e Nome da Disciplina, Codigo e Nome do Curso e a Nota Final.

- 23. Quais alunos forma reprovados, em cada disciplina cursada, por falta (Nnúmero de faltas > 25% do total de aulas). Mostrar: Matricula e Nome do Aluno, Codigo e Nome da Disciplina, Codigo e Nome do Curso, o Numero de Aulas e Número de faltas.
- 24. Quais os professores que reprovaram mais de 3 alunos ( por falta ou por nota). Mostrar o Cpf e o Nome do Professor.
- 25. Qual a média das notas de cada disciplina. Mostrar o Codigo e Nome da disciplina, o código do curso, a soma das notas da disciplina, o número de alunos matriculados e a média.
- 26. Obter a menor e a maior nota obtida em cada disciplina. Mostrar: o Código e Nome da disciplina e a menor e a maior nota.
- 27. Obter a relação dos alunos que obtiveram notas menor que 3,0 e maior que 8,0 em cada disciplina. Mostrar: a Matrícula e Nome do Aluno, Codigo e Nome da disciplina, o Codigo do Curso e a nota obtida.
- 28. Obter a relação das disciplinas de cada curso com os seus respectivos dias de aula. Mostrar: o Codigo e Nome do Curso, Codigo e Nome da Disciplina, O dia de aula (segunda, terça, etc.) , o horário da aula e o tipo (se laboratório ou preleção).
- 29. Obter uma relação com todos os alunos do Departamento de Computação: Mostrar todos os atributos de ALUNO e o Codigo e o Nome do Departamento.
- 30. Obter a relação das disciplinas que não têm alunos matriculados. Mostrar: todos os dados de DISCIPLINA.
- 31. Obter a relação de alunos que não fizeram matrículas.
- 32. Obtenha o número total de alunos reprovados reprovados ((media =( N1\*4 + N2\*6)/10) <m 10) por semestre.
- 33. Obtenha o número total de alunos matriculados em cada curso.
- 34. Encontre a quantidade de aulas lecionadas pelo professor IVON.
- 35. Obter uma relação dos professores do curso de Ciências da Computação, em ordem alfabética. A lista deve conter todos os atributos de Professor e o código do departamento onde ele está vinculado.
- 36. Escreva uma sequencia de comandos que remova todas as disciplinas do curso de Administração. (Lembrar que o Banco de Dados Acadêmico não permite exclusão em cascata)
- 37. Considerando que exista cadastrado no Bancdo Acadêmico o aluno cuja matrícula é 748 escrever o comando que matricula esse aluno na disciplina 7777 do curso de código ADM, na turma A01/1 do primeiro semestre de 1999.
- 38. Listar os alunos, que tiveram notas acima da média de cada turma, em ordem decrescente da nota.
- 39. Listar os alunos e o número de faltas de cada um, em ordem decrescente de nome dos alunos que foram reprovados por falta (25 % do total de aulas dadas).
- 40. Listar as disciplinas que têm aulas na Segunda e na Quarta-feira. A lista deve conter: O codigo e o nome do curso, o codigo, o nome e os créditos da disciplina e o dia da semana);
- 41. Listar os alunos matriculados em Banco de Dados I, eliminando as linhas duplicadas. A lista deve conter, apenas o nome dos alunos;
- 42. Que restrições de integridade podem ser violadas com a execução do comando:

DELETE FROM Professor  
Where Codigo = “FIT”;

- 43. Considerando que o Banco de Dados Acadêmico esteja vazio, por que o comando de inserção abaixo não consegue fazer a inclusão.

INSERT INTO Matrícula  
(Matrícula, Nome, DataMatricula)  
Values (187, “Flavio Rezende Da Matta” , #10-OCT-99#);



44. Criar tabela Disciplina1 com os mesmos dados e requisitos de disciplina.

**Banco de Dados - CIRURGICO**

45. Construir o DER, do banco de dados CIRÚRGICO, que deu origem aos comandos Create Table abaixo:

```
CREATE TABLE Cirurgia
(Número NUMBER (5) NOT NULL,
Data DATE NOT NULL,
Ocorrencias Char (50),
CpfMedico NUMBER (11) NOT NULL,
CpfPaciente NUMBER (11) NOT NULL,
CodigoCirurgia CHAR(3) NOT NULL,
CONSTRAINT PrkCirurgia (Número),
CONSTRAINT FrkTipoCirurgia FOREIGN KEY (CodigoCirurgia)
REFERENCES Tipo_Cirurgia (Codigo),
CONSTRAINT FrkMedicoCirurgia FOREIGN KEY (CpfMedico)
REFERENCES Medico (Cpf),
CONSTRAINT FrkPacienteCirurgia FOREIGN KEY (CpfPaciente)
REFERENCES Paciente (Cpf))

CREATE TABLE Tipo_Cirurgia
(Codigo CHAR (3) NOT NULL,
Nome CHAR (30) NOT NULL,
Peço NUMBER (8) NOT NULL,
CONSTRAINT PrkTipoCirurgia PRIMARY KEY (Codigo));

CREATE TABEL Medico
(Cpf NUMBER(11) NOT NULL,
Nome CHAR(30) NOT NULL,
DataNascimento DATE NOT NULL,
Telefone NUMBER (7),
CONSTRAINT PrkMedico PRIMARY KEY (Cpf));

CREATE TABEL Paciente
(Cpf NUMBER(11) NOT NULL,
Nome CHAR(30) NOT NULL,
DataNascimento DATE NOT NULL,
Telefone NUMBER (7),
Logradouro CHAR (30),
Setor CHAR (20),
Cidade CHAR (20),
Estado CHAR (2),
Cep NUMBER (8),
CONSTRAINT PrkPaciente PRIMARY KEY (Cpf));
```

As questões a seguir têm como referência o Banco de Dados CIRURGICO definido no exercício anterior.

- 46. Escreva o comando para incluir o atributo *TempoDeDuração* na tabela *Cirurgia* do.
- 47. Escreva o comando para excluir o atributo *Ocorrencias* da tabela *Cirurgia*..
- 48. Em que condição ou condições a execução do comando Delete, abaixo, não fere restrições de integridade referencial.

```
DELETE FROM Medico
Where Cpf > 500;
```

49. Em que condição ou condições a execução do comando Update, abiaxo, não fere as restrições de integridade referencial.
- UPDATE Cirurgia

Set CpfPaciente = 458,

CpfMedico = 300

Where Numero = 4;
50. Escreva o comando que reajusta o preço dos tipos de cirurgia em 20%;
51. Listar a quantidade de cirurgias, por tipo, que foram realizadas a partir de 10/10/99;
52. Listar os médicos que não realizaram nenhuma cirurgia no mês de dezembro de 1998;
53. Listar os pacientes que fizeram mais de uma cirurgia no período de 01/02/99 a 30/07/99;
54. Listar os tipos de cirurgia que não tiveram nenhuma cirurgia realizada em 1998;
55. Listar os médicos que realizaram mais de 3 cirurgias no ano de 1999;
56. Listar total de cirurgias por tipo no período de 01/09/99 a 31/12/99;
57. Obter um lista dos médicos em ordem alfabética;
58. Obter uma relação com o cpf e o nome do médico, o número e a data da cirurgia o nome e o preço da cirurgia, o cpf e o nome do paciente. A relação deve ser obtida por ordem alfabética crescente do nome do médico.
59. Listar os pacientes que fizeram cirurgia do coração (Codigo = 15);
60. Listar os tipos de cirurgias, em ordem crescente de código, que não teve nenhum paciente operado no período de 01/01/99 a 01/07/99;

**Banco de dados – FAMILIA ZACHARIAS**

61. Construir o DER que deu origem aos comandos DDL, abaixo:

```
CREATE TABLE Cidade
    Estado CHAR(2) NOT NULL,
    Cidade CHAR(20) NOT NULL,
    CONSTRAINT PrkCidade PRIMARY KEY (Estado, Cidade))

CREATE TABLE Cemiterio
    (Numero NUMBER(5) NOT NULL,
    Estado CHAR(2) NOT NULL,
    Cidade CHAR(20) NOT NULL,
    CONSTRAINT PrkCemiterio PRIMARY KEY (Numero)
    CONSTRAINT FrkCidadeCemiterio FOREIGN KEY (Estado, Cidade)
    REFERENCES Cidade (Estado, Cidade));

CREATE TABLE Pessoa
    (Codigo NUMBER(5) NOT NULL,
    Nome CHARr(30) NOT NULL,
    DataNascimento DATE NOT NULL,
    Sexo CHAR(1) NOT NULL,
    NumeroPaternidade NUMBER(11),
    EstadoNatal CHAR(2) NOT NULL,
    CidadeNatal CHAR(20) NOT NULL,
    Logradouro CHAR(30) NOT NULL,
    Setor CHAR(20) NOT NULL,
    Cidade CHAR(20) NOT NULL,
    Estado CHAR(2) NOT NULL,
    CONSTRAINT PrkPessoa PRIMARY KEY (Codigo),
    CONSTRAINT FrkFilho FOREIGN KEY (NumeroPaternidade)
    REFERENCES Paternidade (Numero),
    CONSTRAINT FkrCidadeNatal FOREIGN KEY (EstadoNatal, CidadeNatal)
    REFERENCES Cidade (Estado, Cidade));

CREATE TABLE Falecimento
    (CodigoPessoa NUMBER(5) NOT NULL,
    DataFalecimento DATE NOT NULL,
    NumeroCertidaoObto NUMBER(8),
    NumeroCemiterio NUMBER(5) NOT NULL,
    CONSTRAINT PrkFalecimento PRIMARY KEY (CodigoPessoa),
    CONSTRAINT FrkPessoaFalecimento FOREIGN KEY (CodigoPessoa)
    REFERENCES Pessoa (Codigo),
    CONSTRAINT FrkCemiterioFalecimento FOREIGN KEY (NumeroCemiterio)
    REFERENCES Cemiterio (Numero));

CREATE TABLE Casamento
    ( Numero NUMBER (11) NOT NULL,
    CodigoPai NUMBER(5),
    CodigoMae NUMBER(5),
    DataCasamento DATE NOT NULL,
    CONSTRAINT PrkPaternidade PRIMARY KEY (Numero),
    CONSTRAINT FrkPai FOREIGN KEY (CodigoPai)
    REFERENCES Pessoa (Codigo),
    CONSTRAINT FrkMae FOREIGN KEY (CodigoMae)
    REFERENCES Pessoa (Codigo));

ALTER TABLE Pessoa
    ADD CONSTRAINT FrkPaternidadePessoa FOREIGN KEY (NumeroPaternidade)
    REFERENCES Paternidade (Numero);
```

Tomando como referência o banco de dados da **FAMILIA ZACHARIAS**, do exercício anterior, resolver as questões a seguir:

- 62. Listar todas as pessoas com pai e mãe desconhecidos. A lista deve conter todos os atributos da pessoa.
- 63. Listar todas as pessoas já falecidas, da família. A lista deve conter: Todos os dados da pessoa, a cidade e o estado onde a pessoa faleceu, a data do falecimento e o número do atestado de óbito.
- 64. Listar as pessoas da cidade de São Paulo que ainda estão vivas e que tenham mais de 70 anos. A lista deve conter todos os atributos de pessoa e deve estar em ordem crescente de Nome da pessoa.
- 65. Listar as pessoas que nasceram no ano de 1998. A lista deve conter todos os atributos de pessoa e deve estar em ordem crescente de data de nascimento.
- 66. Listar as pessoas cuja idade está acima da média de idade da família. A lista deve conter todos os atributos da pessoa, a idade e média de idade da família.
- 67. Obter uma lista das cidades onde mora alguém da família, indicando a quantidade de pessoas da família que mora na cidade, o estado e o nome da cidade.
- 68. Obter uma lista que contenha a quantidade de pessoas falecidas em cada cidade do estado do Paraná.
- 69. Obter uma relação das cidades onde residem pessoas da família. A lista deve conter todos os atributos da pessoa. Ordená-la por ordem crescente de nome da cidade.
- 70. Obter uma relação, em alfabética crescente de nome, das pessoas que têm mais de 3 filhos. A lista deve conter todos os atributos da pessoa.
- 71. Obter a quantidade de homens e de mulheres da família Zacharias.
- 72. Obter uma relação das pessoas que têm mais filhos homens. Listar todos os atributos da pessoa mais a quantidade de filhos e de filhas que cada pessoa tem.
- 73. Listar as pessoas que não têm nenhum filho. A lista deve conter todos os atributos de pessoa.
- 74. Listar as pessoas que são casadas. A lista deve conter o nome do esposo, o nome da esposa e da data de casamento, e deve ser ordenada por nome do esposo (ascendente).
- 75. Listar os casais que não têm filhos.
- 76. Listar as pessoas solteiras.
- 77. Listar as pessoas que tiveram mais de um casamento.

**Banco de Dados – TRANSPORTE - de uma empresa transportadora de mercadorias**

78. Construir o DER que deu origem aos comandos relacionados a seguir:

```
CREATE TABLE Cliente
    (Codigo NUMBER (5) NOT NULL,
    Nome Char(30) NOT NULL,
    Cpf NUMBER(11),
    Cgc NUMBER (14),
    Telefone NUMBER (7),
    E-Mail CHAR (20),
    Logradouro CHAR(30),
    Setor CHAR(20),
    Cidade CHAR(20),
    Estado CHAR(20),
    Cep NUMBER (8);
    CONSTRAINT PrkCliente PRIMARY KEY (Codigo));

CREATE TABLE Frete
    (Numero NUMBER (8) NOT NULL,
    DataDeEmbarque DATE NOT NULL,
    ValorDaMercadoria NUMBER (7) NOT NULL,
    PesoDaMercadoria NUMBER (3) NOT NULL,
    ValorDoFrete NUMBER (7) NOT NULL,
    CodigoRemetente NUMBER (5) NOT NULL,
    CodigoDestinatario NUMBER (5) NOT NULL,
    EstadoOrigem CHAR (2) NOT NULL,
    CidadeOrigem CHAR (20) NOT NULL,
    EstadoDestino CHAR (2) NOT NULL,
    CidadeDestino CHAR (20) NOT NULL,
    CONSTRAINT PrkFrete PRIMARY KEY (Numero),
    CONSTRAINT FrkRemetente FOREIGN KEY (CodigoRemetente)
        REFERENCES Cliente (Codigo),
    CONSTRAINT FrkDestinatario FOREIGN KEY (CodigoDestinatario)
        REFERENCES Cliente (Codigo),
    CONSTRAINT FrkOrigem FOREIGN KEY (EstadoOrigem,CidadeOrigem)
        REFERENCES Cidade (Estado,Cidade),
    CONSTRAINT FrkDestino FOREIGN KEY (EstadoDestino,CiddeDestino)
        REFERENCES Cliente (Estado,Cidade));

CREATE TABLE Cidade
    (Estado Char(2) NOT NULL,
    Cidade Char(20) NOT NULL,
    CONSTRAINT PrkCidade PRIMARY KEY (Estado, Cidade),
    CONSTRAINT FrkEstadoCidade FOREIGN KEY (Estado)
        REFERENCES Estado (Estado));
```

```
CREATE TABLE Estado
    (Estado Char(2) NOT NULL,
     Nome Char(20) NOT NULL,
     CONSTRAINT PrkEstado PRIMARY KEY (Estado));

CREATE TABLE Icms
    (EstadoOrigem Char(2) NOT NULL,
     EstadoDestino Char(2) NOT NULL,
     Taxa NUMBER(5) NOT NULL,
     CONSTRAINT PrkIcms PRIMARY KEY (EstadoOrigem, EstadoDestino)
     CONSTRAINT FrkEstadoOrigemIcms FOREIGN KEY (EstadoOrigem)
     REFERENCES Estado (Estado),
     CONSTRAINT FrkEstadoDestinoIcms FOREIGN KEY (EstadoDestino)
     REFERENCES Estado (Estado));
```

Tendo como referência o Banco de Dados TRANSPORTE, do exercício anterior, responder as questões a seguir:

- 79. Escreva uma sequencia de comandos que provoque uma violação de integridade de entidade, na tabela Cliente.
- 80. Escreva uma sequencia de comandos que provoque uma violação de integridade referencial entre as tabelas Frete e Cliente.
- 81. Escreve um comando que provoque uma violação de integridade de domínio, na tabela Cliente.
- 82. Escreva um comando que provoque uma violação de integridade de chave, na tabela Estado.
- 83. Listar os fretes das mercadorias embarcadas no período de 25/10/99 a 24/11/99. A lista deve conter, além dos atributos do frete, o Nome do remetente e do destinatário da mercadoria.
- 84. Listar o valor do menor e do maior frete transportado no ano de 1998. A lista deve conter, além dos atributos do frete, o Nome do remetente e do destinatário da mercadoria.
- 85. Listar os clientes que não enviaram e nem receberam mercadorias no ano de 1998; A lista deve conter o Codigo, Nome, Cpf, Cgc do Cliente e deve ser mostrada em orderm crescente de nome do cliente.
- 86. Listar os clientes cujos fretes pagos estão acima da média dos fretes transportados no primeiro trimestre de 1999.
- 87. Obter a quantida de fretes, enviados, para cada cliente no segundo semestre de 1999. Listar, apenas, aqueles clientes que transportaram mercadorias mais de cinco vezes no período.
- 88. Qual o Estado ou estados que têm a maior taxa de Icms para fora do estado.
- 89. Quantas vezes uma cidade recebeu mercadorias transportadas. Listar, apenas, aquelas que receberam mais de 20 vezes. A listar deve estar em ordem decrescente de nome da cidade.
- 90. Escreva um comando para adicionar o atributo Sexo em Cliente;
- 91. Escreva um comando para retirar o atributo Cgc de Cliente;
- 92. Escreva um comando que cria uma tabela Frete1 com as mesmas características e que contenha os mesmos dados da tabela Frete;
- 93. Listar o Cliente que pagou o maior frete à transportadora;
- 94. Escrava um comando para criar uma tabela Estado1 com atributos sufixados de 1, com as mesmas características e que contenha os mesmos dados da tabela Estado;
- 95. Listar os estados de onde originou mais de 40% dos valores dos fretes pagos.

**Banco de Dados – ACADÊMICO – Parte II**

1. Listar todos os professores em ordem alfabética descendente de nome. A lista deve conter todos os atributos de professor.
2. Listar todos os professores, do departamento de computação, em ordem descendente de nome. A lista deve conter todos os atributos de professor.
3. Listar a quantidade de professores vinculados a cada departamento. A lista deve conter o código do departamento, a quantidade de professores do departamento e ser ordenada por código do departamento em ordem descendente.
4. Listar os departamentos que têm mais de 5 professores a ele vinculados. A lista deve conter o código do departamento, a quantidade de professores do departamento e ser ordenada por código do departamento em ordem crescente.
5. Listar os professores com mais de vinte e cinco anos. A lista deve conter o Cpf, o nome, a data de Nascimento e a idade de cada professor. A lista deve ser ordenada por ordem crescente de idade.
6. Listar os professores que não têm endereço cadastrado. A lista deve conter todos os atributos de professor e ser ordenada em ordem crescente de nome.
7. Obter uma lista de todos os cursos em ordem crescente de código do curso. A lista deve conter todos os atributos de curso.
8. Listar a quantidade de cursos vinculados a cada departamento. A lista deve conter o código do departamento e a quantidade de cursos a ele vinculada. A lista deve ser ordenada em ordem decrescente da quantidade de cursos obtida.
9. Listar os departamentos com mais de 2 cursos a ele vinculados. A lista deve conter o código do departamento e a quantidade de cursos a ele vinculada. A lista deve ser ordenada em ordem decrescente da quantidade de cursos obtida.
10. Obter uma lista das disciplinas do curso de Ciências da Computação. A lista deve conter todos os atributos de disciplina e deve ser mostrada em ordem crescente de código do curso e código da disciplina.
11. Listar as disciplinas do curso de Administração que têm menos de 4 créditos. A lista deve conter todos os atributos de disciplina e deve ser mostrada em ordem crescente de créditos.
12. Obter a carga horária do curso de Ciência da Computação. Listar o código do curso e a carga horária em ordem crescente carga horária.
13. Listar as disciplinas que não têm aulas de laboratório. A lista deve conter todos os atributos da disciplina e deve ser mostrada em ordem decrescente de código do curso e código da disciplina.
14. Obter a quantidade de disciplinas, a quantidade de créditos, a quantidade de aulas de preleção e a quantidade das aulas de laboratório de cada curso. A lista deve conter além das quantidades pedidas o código do curso e ser ordenada por código do curso.
15. Obter uma lista dos cursos cuja carga horária sejam maior que 400 horas. Listar o código do curso e a carga horária e ordenar a lista por código do curso.
16. Listar as disciplinas oferecidas no ano de 1999. A lista deve conter o código do curso, o código da disciplina e deve ser ordenada por código do curso e código da disciplina em ordem crescente.
17. Obter uma lista dos alunos em ordem crescente de nome. A lista deve conter todos os atributos de aluno.



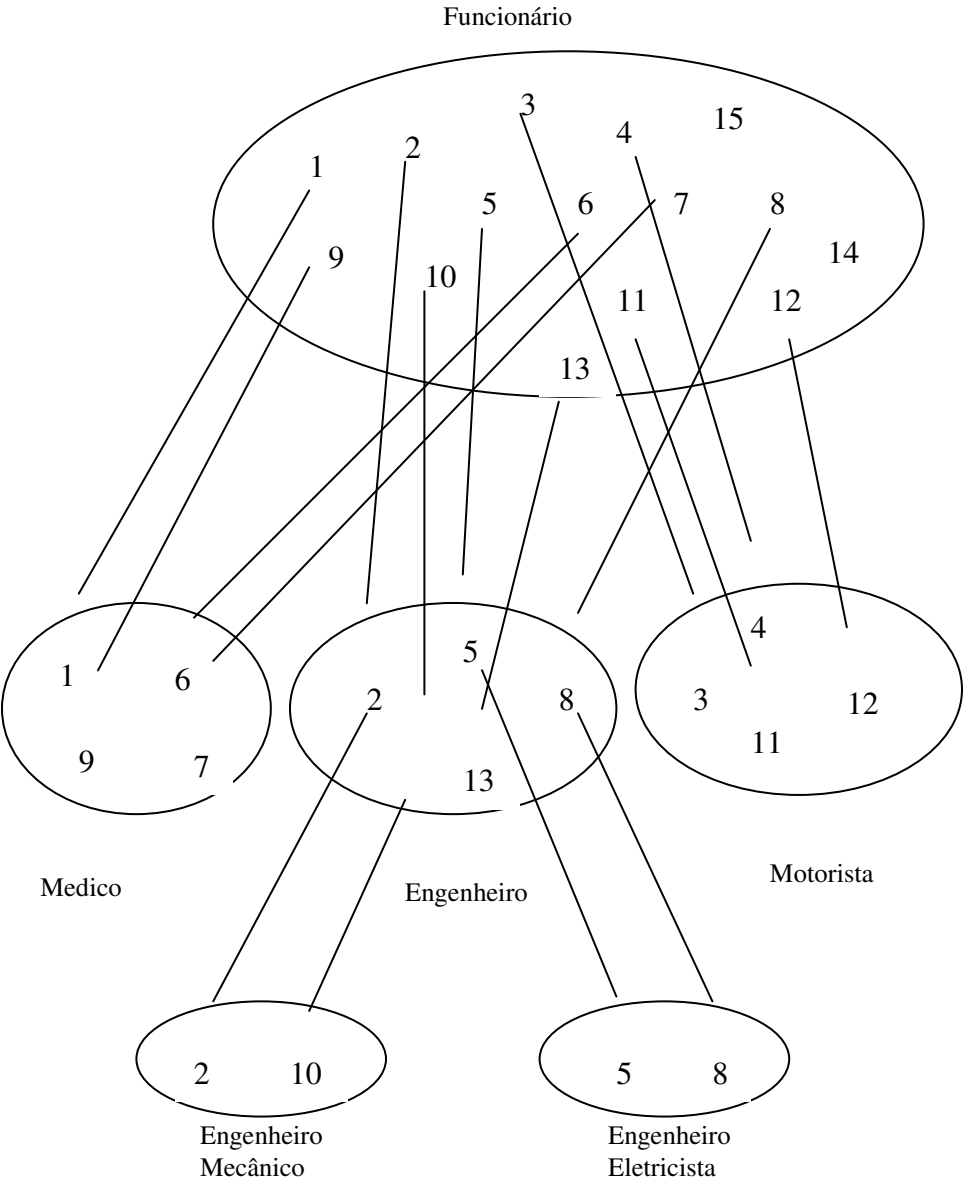
- 18. Obter uma lista dos alunos que não têm telefone. A lista deve conter todos os atributos de aluno e deve ser ordenada por nome do aluno em ordem decrescente.
- 19. Obter uma lista com a média geral de cada disciplina, no primeiro semestre de 1999. A lista deve conter o código do curso, o código da disciplina e a média e deve ser mostrada em ordem decrescente da média.
- 20. Obter uma lista das disciplinas cuja média geral, no segundo semestre de 1999 tenha ficado inferior a 5. A lista deve conter o código do curso, o código da disciplina e a média e deve ser mostrada em ordem decrescente da média.
- 21. Obter uma relação dos cursos cuja média geral no primeiro semestre de 1999 seja superior a 6. A lista deve conter o código do curso e a média e estar ordenada por código do curso.
- 22. Obter o total de alunos matriculados em cada subturma aberta no primeiro semestre de 1999. A lista deve conter o código do curso, o código da disciplina, o código da turma e da subturma, o ano e o semestre e o total obtido.
- 23. Listar os alunos reprovados em Banco de Dados I (nota < 5,0). A lista deve conter a matrícula do aluno, o código da disciplina, o código do curso e a nota do aluno. Mostrar a lista em ordem crescente de código do curso e código da disciplina.
- 24. Obter uma lista das turmas abertas no segundo semestre de 1998 com menos de 10 alunos matriculados. A lista deve conter o código do curso, o código da disciplina, o código da turma e da subturma, o ano e o semestre e o total obtido. Mostrar a lista em ordem crescente de ano, semestre, código do curso, código da disciplina.
- 25. Listar as matrículas ocorridas entre 01/01/99 e 30/01/99. A lista deve conter todos os atributos de matrícula e deve ser mostrada em ordem crescente de matrícula, turma e subturma.
- 26. Listar a quantidade de aulas acontecidas entre 01/03/99 e 30/04/99.
- 27. Listar a quantidade de aulas acontecidas no primeiro semestre de 1999, por curso. A lista deve conter o código do curso e a quantidade de aulas obtidas. Ordenar a lista em ordem decrescente de código de curso.
- 28. Listar a quantidade de aulas acontecidas no primeiro semestre de 1999, por disciplina. Mostrar na lista o código do curso, o código da disciplina e a quantidade de aulas obtidas. Ordenar a lista em ordem crescente de código de curso e código de disciplina.
- 29. Listar as disciplinas que tiveram mais de 16 aulas por mês. A lista deve conter o código do curso, o código da disciplina, o mes e a quantidade de aulas obtidas. Ordenar a lista em ordem crescente de código de curso e código da disciplina.
- 30. Obter uma lista dos alunos reprovados por falta (faltas >= 25 % das aulas dadas). A lista deve estar ordenada por matrícula, em ordem crescente, e deve conter a matrícula do aluno, o código do curso, o código da disciplina, o código da turma, o código da subturma, o semestre e o ano, o total de faltas e o total de aulas dadas.
- 31. Listar o total de aulas assistidas por cada aluno da turma A01, subturma 2 da disciplina CMP 1010 no segundo semestre de 1999. Mostrar a matrícula e o total de aulas assistidas em ordem crescente de matrícula.

- 32. Obter o total de disciplinas do curso de Ciências da Computação. Mostrar na lista o código e o nome do curso e o total de disciplinas obtido. A lista deve ser mostrada em ordem crescente de código de curso.
- 33. Obter a quantidade de turmas aberta por disciplina no semestre. Mostrar na lista, o código e o nome do curso, o código e o nome da disciplina, o ano e o número do semestre e a quantidade de turma obtida, e ordená-la por ano, semestre, código e nome do curso e código e nome da disciplina.
- 34. Listar as disciplinas vinculadas a cada professor. A lista deve conter o cpf e o nome do professor, o código e o nome do curso e o código e o nome da disciplina. Ordenar a lista em ordem crescente de: Nome do professor, código do curso e código da disciplina.
- 35. Obter uma relação dos professores que deram aula no primeiro semestre de 1999. A relação deve conter os seguintes dados: cpf e nome do professor, código e nome do curso, código e nome da disciplina e a data da aula. Ordenar a relação por nome do professor, código do curso e código da disciplina, em ordem crescente.
- 36. Relacionar a quantidade de aulas dadas por cada professor no ano de 1999. A relação deve conter o nome do professor e a quantidade de aulas dadas e deve estar ordenada por nome do professor em ordem crescente.
- 37. Relacionar os alunos de cada curso em ordem crescente de código e nome do curso e nome do aluno. A relação deve conter o código e o nome do curso e a matrícula e o nome do aluno.
- 38. Listar os alunos matriculados no curso de Filosofia, por semestre, a partir de 1997. Mostrar a matrícula e o nome do aluno, o código e o nome do curso, o ano e o semestre. Ordenar em ordem crescente de ano, semestre, nome do curso e nome do aluno.
- 39. Listar o resumo das aulas, de cada professor, do primeiro semestre de 1998. A lista deve conter o nome do professor, o código do curso, o código da disciplina, o número da aula, o código da turma, a subturma, a data da aula e o resumo. Ordenar por nome do professor, código da turma, subturma e data da aula, em ordem crescente.
- 40. Obter uma relação de alunos com as respectivas aulas assistidas, do segundo semestre de 1999. Mostrar a matrícula e o nome do aluno, o código e o nome do curso, o código e o nome da disciplina, a turma, subturma e a data da aula. Ordená-la por nome do aluno, nome do curso, nome da disciplina e data da aula.
- 41. Relacionar os alunos reprovados por falta (faltas >= 25 % das aulas dadas) no primeiro semestre de 1999. Mostrar na relação a matrícula e o nome do aluno, o código do curso, o código e o nome da disciplina, a quantidade de aulas dadas, a quantidade de faltas e o percentual das faltas em relação às aulas dadas. Ordená-la por nome do aluno, por código do curso e código da disciplina.
- 42. Relacionar os alunos reprovados por nota (< 5,0) no segundo semestre de 1998. A relação deve conter os seguintes dados: matrícula e nome do aluno, código do curso, código e nome da disciplina e a nota. Mostrar a relação ordenada por: matricula do aluno, código do curso e código da disciplina.
- 43. Obter uma relação dos alunos reprovados por um professor dado, em um semestre dado de um ano dado. Apresentar na relação o ano, o semestre e o cpf dados, o nome do professor, o código do curso, o código e o nome da disciplina, a matrícula e o nome do aluno. Classificar a relação por: nome do aluno, ano, semestre, nome do professor.

44. Listar quantas vezes um determinado aluno foi reprovado numa determinada disciplina. Mostrar a matrícula e o nome do aluno, o código do curso e o código e o nome da disciplina. Ordenar a lista por nome do aluno, código do curso e código da disciplina.
45. Listar quantas vezes um determinado aluno cursou determina disciplina. Mostrar a matrícula e o nome do aluno, o código do curso e o código e o nome da disciplina. Ordenar a lista por nome do aluno, código do curso e código da disciplina.
46. Obter uma relação dos professores que não lecionaram em 1998. Mostrar na relação todos os dados do professor e ordená-la por nome do professor.
47. Obter uma relação das disciplinas que não foram oferecidas num determinado semestre. A relação deve conter: o ano, o semestre, o código do curso e o código e o nome da disciplina e deve ser ordenada por ano, semestre, código do curso e nome da disciplina, em ordem crescente.
48. Listar os alunos que não matricularam em 1997 em nenhuma disciplina. A ordenação e o conteúdo da lista deve ser sugerido pelo aluno.
49. Listar as disciplinas que tiveram mais de 20% de reprovação no primeiro semestre de 1999. A ordenação e o conteúdo da lista deve ser sugerido pelo aluno.
50. Obter uma lista de aulas que tiveram menos de 15 alunos presentes. Mostrar na lista o nome e a matrícula do aluno, o código do curso, o código e o nome da disciplina e a quantidade de alunos obtida. Sugerir a classificação mais interessante para a lista.
51. Obter uma lista dos alunos com nota acima da média da turma. Mostrar: a matrícula e o nome do aluno, o código do curso, código e nome da disciplina, o código da turma, a média da turma e a nota do aluno. Ordenar a lista por nome do aluno, código do curso e nome da disciplina.
52. Listar os professores que deram aula nas Terças-feiras de 1998 . Mostrar o nome do professor e o código e o nome das disciplinas lecionadas e o código da turma e a subturma. Ordenar a lista por nome do professor.
53. Listar os alunos que só assistiram 70% das aulas do mês de marco do primeiro semestre de 1998. Sugerir o conteúdo e ordenação que sejam mais interessantes para a lista.
54. Obter uma lista dos alunos cujo número de faltas seja maior que o número de faltas da turma. Sugerir o conteúdo e a ordenação mais interessantes para a lista.

EXERCÍCIOS DIVERSOS

1. A figura abaixo mostra a correspondência entre as entidades FUNCIONÁRIO, ENGENHEIRO, MOTORISTA, MÉDICO, ENGENHEIRO MECÂNICO E ENGENHEIRO ELETRICISTA do modelo de dados do Sistema de Recursos Humanos de uma empresa. Construir o DER correspondente as entidades descritas. Mostrar, no DER, apenas as entidades e as cardinalidades de cada relacionamento.



2. O segmento de relatório abaixo, é produzido por um sistema hipotético de controle de voos aéreos domésticos nacionais. Construir o segmento de DER do modelo de dados do sistema que possa ter dado origem ao relatório mostrado.

Empresa	Voo	Rota				
Varig	150	Goiânia – GO Chegada: 06:10 Partida: 07:10	São Paulo – SP Chegada: 07:10 Partida: 08:00	R. Janeiro –  Chegada: 09:00 Partida: 09:45	Salvador – BA Chegada: 11:45 Partida: 12:00	Aracaju – SE Chegada: 12:30 Partida: 12:45
Varig	151	São Paulo – SP Chegada: 12:10 Partida: 12:30	Curitiba – PR Chegada: 13:30 Partida: 14:00	Florianópolis – SC Chegada: 15:00 Partida: 15:30	Porto Alegre –  Chegada: 16:30 Partida: 17:00	
Vasp	152	R. de Janeiro –  Chegada: 07:00 Partida: 08:00	Florianópolis –  Chegada: 09:00 Partida: 09:30	São Paulo – SP Chegada: 11:10 Partida: 12:00	Goiânia – GO Chegada: 13:10 Partida: 13:30	Brasília – DF Chegada: 13:50 Saída: 14:00
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
TAM	200	Uberlândia – Mg Chegada: 10:00 Saída: 11:00	São Paulo – SP Chegada: 12:00 Partida: 12:30	Brasília – DF Chegada: 13:00 Saída: 14:00		

3. Um hospital precisa saber ao final de cada mês, para efeito de pagamento dos médicos. O pagamento dos médicos é feito com base em um relatório que é emitido no final do mês que contém os seguintes dados: nome e endereço de cada paciente operado. Nome de cada médico que fez a cirurgia no paciente. Qual médico foi o responsável e quais foram os médicos auxiliares de cada cirurgia. O nome de cada cirurgia e o valor que deve ser pago a cada médico por cirurgia e o valor total que deve ser pago para cada médico por mês. Há pacientes que fazem mais de um cirurgia.

Construir o DER de um sistema que possa atender ao hospital.

4. O segmento de relatório abaixo mostra parte da árvore genealógica de uma família. Construir o DER a partir do qual seja possível emitir o relatório. As colunas correspondentes a Pai e Mãe não preenchidos significam que a pessoa tem pai e/ou mãe desconhecidos.

Pessoa		Pai		Mãe	
Codigo	Nome	Codigo	Nome	Codigo	Nome
...	...	...	...	...	...
001	Rafael Ribeiro dos Anjos	007	João Ribeiro	010	Maria dos Anjos
002	Ulisses Resende dos Santos	008	Francisco Resende	011	Rosa dos Santos
003	Maria das Graças Ribeiro	001	Rafael Ribeiro dos Anjos	009	Carla Mendes
004	Maria Ribeiro dos Anjos	007	João Ribeiro	010	Maria dos Anjos
005	João Martins Filho				
006	Carlos Gomes				
007	João Ribeiro				
008	Francisco Resende				
009	Carla Mendes				
010	Maria dos Anjos	012	Rogério dos Anjos		
011	Rosa dos Santos				
012	Rogério dos Anjos				
...	...	...	...	...	...

5. A lista abaixo, mostra o conteúdo dos atributos de entidades de um tipo de entidade A. Existe algum atributo, da relação, que pode ser escolhido como identificador da entidade? Justifique sua resposta.

A	Nome	Endereço	Sexo	Data	Estado Civil
	Francisco dos Anjos	Rua 18	M	10/10/47	Viúvo
	Maria Antonieta	Rua 17	F	15/08/50	Desquitado
	Carlos Rosemberg	Rua 14	M	20/03/60	Divorciado
	Flacio Rocha	Rua 18	F	10/10/47	Viúvo
	Maria Antonieta	Rua 12	M	12/12/40	Casado

6. Construir o DER que possa ter dado origem ao comando CREATE TABLE, escrito abaixo. (Representar entidades, relacionamentos e suas cardinalidades e os atributos).

**CREATE TABLE APROVEITAMENTO**

```
(CodigoAluno INTEGER NOT NULL,  
CodigoDisciplina INTEGER NOT NULL,  
Nota BYTE NOT NULL,  
CONSTRAINT PrkAproveitamento  
PRIMARY KEY (CodigoAluno, CodigoDisciplina),  
CONSTRAINT FrkDisciplinaAproveitamento  
FOREIGN KEY (CodigoAluno)  
REFERENCES ALUNO (Codigo),  
CONSTRAINT FrkAlunoAproveitamento  
FOREIGN KEY (CodigoDisciplina)  
REFERENCES DISCIPLINA (Código))
```

7. Construir o DER correspondente ao ambiente descrito abaixo (representar, apenas, as entidades e os relacionamentos e suas cardinalidades):
- 1) Para cada Cliente pode ser emitido várias Notas Fiscais;
  - 2) Cada Nota Fiscal deve se referir a um único Cliente;
  - 3) Cada Nota Fiscal deve ser composta de vários Itens de Nota Fiscal;
  - 4) Cada Item de Nota Fiscal deve estar associado a uma única Nota Fiscal;
  - 5) Cada Item de Nota Fiscal deve se referir a um único Produto;
  - 6) Cada Produto pode estar associado a vários Itens de Nota Fiscal.

8. Construir o Diagrama de Entidade x Relacionamento, explicitando entidades, relacionamentos, identificadores e cardinalidades, correspondente às relações abaixo. As colunas com nomes sublinhadas são as Primary Key da relação.

<b>PEDIDO</b>	<u>Ano</u>	<u>Número</u>	Data de Emissão	Codigo.Fornecedor
	1998	00001	10/10/98	01
	1998	00002	10/11/98	01
	1999	00001	20/01/99	03
	1999	00002	20/02/99	03

<b>PRODUTO</b>	<b>Codigo</b>	Descrição	Saldo	Unidade
	A1	Forno microondas	100	Unitário
	A2	Anzol	800	Duzia
	A3	Adubo orgânico KBX	100	Kg
	A4	Caderno Espiral 100 Fls	180	Unitário

<b>DETALHE DE PEDIDO</b>	Item	Quantidade	<b>Codigo.Produto</b>	<u>Ano.Pedido</u>	<u>Número.Pedido</u>
	01	5	A1	1998	00001
	02	10	A2	1998	00001
	03	3	A3	1998	00001
	01	8	A1	1998	00002
	02	60	A2	1998	00002
	01	20	A3	1999	00001

<b>FORNECEDOR</b>	<b>Codigo</b>	Nome	Endereço	CGC	Capital Social	Categoria
	01	Irmãos Cunha	Rua 15	CGC1	3.000.000,00	Atacadista
	02	Casa da Pesca	Rua 80	CCG2	150.000,00	Varejista
	03	Pecuarista do Povo	Rua São João	CGC3	200.000,00	Rural
	04	Papelaria Moderna	Rua 23	CGC3	100.000,00	Escolar

9. Com base nas relações da questão anterior, obter, através de operações da álgebra relacional, uma relação indicadora do resumo de pedidos de cada produto por ano. A relação deve conter: O ano do pedido, O Código e o Nome do Produto, a Soma das quantidades de cada DETALHE DE PEDIDO para cada produto no ano e o número de ocorrências de cada produto em DETALHE DE PEDIDO no ano.



10. Com base nas relações da *Questão 2*, obter, através de operações da álgebra relacional, uma relação de fornecedores que não têm pedidos a eles relacionados. A relação deve conter todas as colunas da relação FORNECEDOR.
11. Com base nas relações, A e B, abaixo, mostrar o conteúdo da relação obtida através da expressão:  $B - (A \cap B)$

A	Codigo	Nome
	1	Manoel dos Santos
	2	Raquel Queiroz
	4	Francisco Menelau
	5	Ramsés Ribas

B	Codigo	Nome
	5	Ramsés Ribas
	4	Francisco Rezende
	9	Francisco Menelau
	10	Ramsés Ribas

$B - (A \cap B)$	Codigo	Nome

12. Com base nas relações, A e B, abaixo, mostrar o conteúdo da relação obtida através da expressão:  $(A \cup B) \cup B$

A	Codigo	Nome
	00001	Maria Vieira
	00120	Merida França
	00430	Viviane Costa
	00130	Wilmar Cardoso
	00028	Rocha Rezende

B	Codigo	Nome
	00028	Rocha Rezende
	00040	Francisco Rezeck
	00001	Maria Vieira
	00120	Rosa Maria

$(A \cup B) \cup B$	Codigo	Nome

13. Um pessoa ou uma empresa compram veículos financiados através da rede bancária. Uma mesma empresa pode ter vários veículos financiados ao mesmo tempo, inclusive com financiamento de bancos diferentes. Já uma pessoa só pode comprar outro carro financiado depois de quitar o financiamento anterior, se existir. Há um contrato, para cada carro financiado, que é identificado por: um número e pela data de emissão. E mais, o valor, a data do financiamento, o valor da prestação, a quantidade de prestações e a data de vencimento de cada prestação. Cada contrato se refere a um único veículo, mesmo que a mesma pessoa ou empresa tenha financiado mais de um veículo. Cada veículo é identificado por: cor, marca, número do chassi e a quantidade de passageiros. Do agente financeiro é preciso: do nome, do endereço e do CGC. Das empresas há que ser informado: o CGC, a razão social, o nome de fantasia e endereço, além do CPF, do endereço, e do nome de cada um de seus sócios. Uma pessoa, que tenha carro financiado, deve ser identificada pelo nome, endereço, CPF, Número da carteira de identidade e órgão emissor. Para cada contrato de financiamento há pelo menos um avalista, podendo existir mais de um. Há contratos que têm como avalistas empresas e pessoas físicas. Cada avalista é identificado, por CGC, Razão Social e endereço, se empresa, e CPF, nome e endereço, se pessoa física.

Construir um DER para o ambiente acima, mostrando: entidades com seus atributos, identificador de cada entidade e entidades fracas.

14. Dado as duas relações abaixo mostre a relação RESULTADO obtida da operação  $A - B$ .

A	Matricula
	1
	2
	3
	4
	5
	6
	7

B	Matricula
	1
	3
	2
	4

RESULTADO	

15. Dado as relações A e B abaixo, mostrar o conteúdo da relação C após as a sequência de operações a seguir:

$$C \leftarrow (A \cup B) \cap B$$

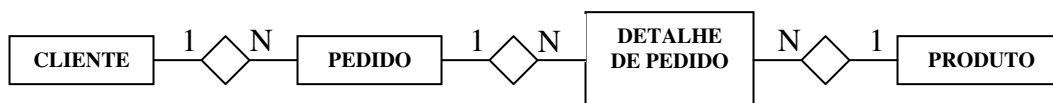
A	Matricula
	1
	2
	3
	4
	5
	6
	7

B	Matricula
	1
	3
	2
	4

C	Matricula

16. Dado o DER e a descrição de seus atributos, abaixo, escrever os dois comandos SELECT pedidos a seguir:

- a) **(1,5 pontos)** - Listar a quantidade de pedidos de cada cliente. A lista deve conter o código e o nome do cliente e o número. Mostrar a lista em ordem decrescente da quantidade obtida.
- b) **(1,5 pontos)** - Listar os pedidos de cada cliente. A lista deve conter o número e a data do pedido, o código e o nome do produto, o código e o nome do cliente e a quantidade de cada detalhe de pedido de cada produto.



CLIENTE	Atributos	CP	CE	TIPO	TAM
	Codigo	S		Number	3
	Nome			Text	30
	DataNascimento			Date	

PEDIDO	Atributos	CP	CE	TIPO	TAM
	Numero	S		Number	5
	Data			Date	30
	Valor			Number	8
	CodigoCliente		S		

DETALHE DE PEDIDO	Atributos	CP	CE	TIPO	TAM
	NumeroPedido	S	S		
	Item	S		Number	2
	Quantidade			Number	5
	CodigoProduto		S		

PRODUTO	Atributos	CP	CE	TIPO	TAM
	Codigo	S		Text	3
	Nome			Text	30
	Valor			Number	8

Legendas: CP = Chave Primária da tabela;

CE = Chave Estrangeira

TIPO = Tipo de Dado

TAM = Tamanho do campo

17. O serviço de transporte urbano de pessoas, em Goiânia, é executado pelas empresas de transporte coletivo sob o regime de concessão do Governo Estadual. A ARAGUAIA e HP são algumas das empresas que prestam esse tipo de serviço. O sistema funciona com base no conceito de linhas. Uma linha representa o trajeto que um ônibus deve percorrer – quais ruas por onde ele deve passar - para prestar os seus serviços. A cada linha criada e concedida é dado um número, que é único para todo o sistema, e a data de sua concessão e é definido seu trajeto. Cada rua do trajeto contém os pontos de ônibus onde os usuários to transporte toma o ônibus. A cada rua é dado um número, que é único para o sistema, e o nome, e a cada ponto de ônibus é dado um número, que também é único para o sistema, e a ordem dele dentro da rua (primeiro, segundo..., etc). Construir o diagrama de entidades x relacionamentos que mostre as linhas, os pontos de ônibus, pelos quais a linha passa, e as ruas do sistema. Apresentar a matriz de definição dos atributos.
18. Construir o DER, do banco de dados relacional, correspondente à sequência de comandos abaixo:

```
a) CREATE TABLE AUTOR
      (Cpf Long NOT NULL,
       Nome Text(30) NOT NULL,
       Fone Long,
       CONSTRAINT PrkCpf PRIMARY KEY (Cpf));

CREATE TABLE EDITORA
      (Codigo Long NOT NULL,
       Nome Text(30) NOT NULL,
       Fax Long,
       E-Mail Text(50),
       CONSTRAINT PrkEditora PRIMARY KEY (Codigo);

CREATE TALBE LIVRO
      (Numero Long NOT NUL,
       Titulo TEXT(30) NOT NULL,
       Assunto TEXT(30) NOT NULL,
       CpfAutor Long NOT NULL,
       Editora Long NOT NULL,
       CONSTRAINT PrkLivro PRIMARY KEY (Numero),
       CONSTRAINT al FOREIGN KEY (CpfAutor)
           REFERENCES AUTOR (Cpf),
       CONSTRAINT el FOREIGN KEY (Editora)
           REFERENCIES EDITORA(Codigo));
```

- b) Considerando que o banco, correspondente aos comandos do *item a*, esteja criado, o que acontece com a execução do comando abaixo. Justifique sua resposta.

```
DROP TABLE AUTOR;
```

19. Para a resolução das questões propostas a seguir, considerar o banco de dados criado na *questão anterior, item a*.

a) Escreva a sequência de comandos, na ordem correta, para inserir duas linhas em cada tabela.

b) (Em que condições o comando, abaixo, pode provocar uma violação de integridade referencial?

```
DELETE FROM AUTOR
WHERE Cpf < 11111111111;
```

c) Escreva uma sequência de comandos que provoque uma violação de integridade de chave, na tabela AUTOR.

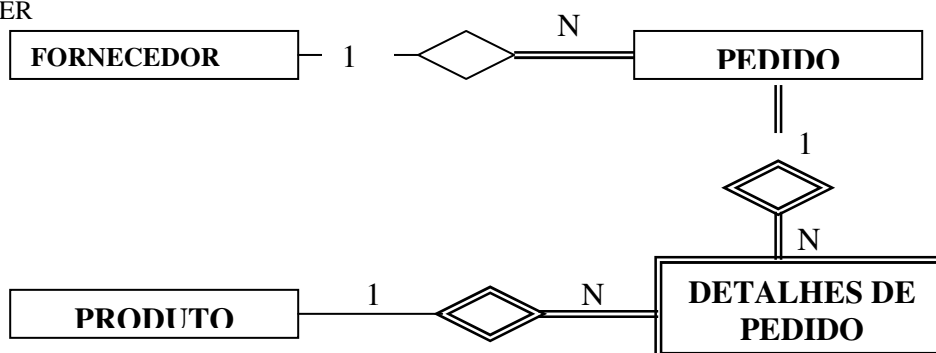
20. Para a resolução das questões propostas a seguir, considerar o banco de dados criado na *questão 18 item a*.

a) Escreva o comando para listar os autores que não tenham nenhum livro editado. A lista deve conter todos os dados de Autor;

b) Listar os autores com livros editados. A lista deve conter todos os dados de autor, o número e o título do livro e o código e o nome da editora. Ordenar a lista por nome da editora, nome do autor e título do livro.

21. As questões 22 e 23 a seguir devem ser respondidas de acordo com as definições do Banco de dados abaixo:

1.a. DER



1.b. Definição dos atributos

FORNECEDOR	Atributo	Id	Relacionamento	Tipo	Req	Tam	Dec	Dom
	Código	S		N	S	3	0	> 0
	Nome			Text	S	40		Qualquer
	Cgc			N		14	0	> 0
	Fone			N		7	0	> 999999

PEDIDO	Atributo	Id	Relacionamento	Tipo	Req	Tam	Dec	Dom
	Numero	S		N	S	4	0	> 0
	Data			Date	S	8		> 31/12/00
	Logradouro			Text		40		Qualquer
	Bairro			Text		20		Qualquer
	Cidade			Text		20		Qualquer
	Uf			Text		2		Sigla dos Estados
	Cep			N		8		> 0
	CodForn		Fornecedor-Pedido	N	S	3		>0

DETALHES DE PEDIDO	Atributo	Id	Relacionamento	Tipo	Req	Tam	Dec	Dom
	NumPedido	S	Pedido-Detalhes de Pedido	N	S	4	0	> 0
	CodProduto	S	Produto-Detalhes de Pedido	Text	S	4		Qualquer
	Quantidade			N	S	6	1	>=0
	Desconto			N		5	2	>=0 e <= 100

PRODUTO	Atributo	Id	Relacionamento	Tipo	Req	Tam	Dec	Dom
	Código	S		Text	S	4		Qualquer
	Nome			Text	S	30		Qualquer
	Preço			N	S	8	2	>= 0,10

22. Considerando que as tabelas abaixo são tabelas do banco de dados relacional definido anteriormente, encontrar pelo menos sete erros de inconsistência de dados das tabelas mostradas. Justificar a sua resposta.

FORNECEDOR	Colunas			
Linhas	Código	Nome	Cgc	Fone
01	001	Casa do Lavrador	10101010000101	2000000
02	002	Fornecedor X	8181818100023459	
03	003	Fornecedor Y		
04	004	Fornecedor Z	191919191911	4000000

PEDIDO	Colunas							
Linhas	Número	Data	Logradouro	Bairro	Cidade	Uf	Cep	CodForn
01	0001	10/10/87	Rua 10	Centro	Goiania	Go	74000000	001
02	0002	01/01/01	Rua 13			Ms	32001030	010
03	0003	01/01/51						001
04	0020	30/70/70	Rua 55	S Sul	Itumbiara	Rs	00001000	001
05		10/10/02	Rua 70	S Oeste	Goiânia	Go	00002000	004

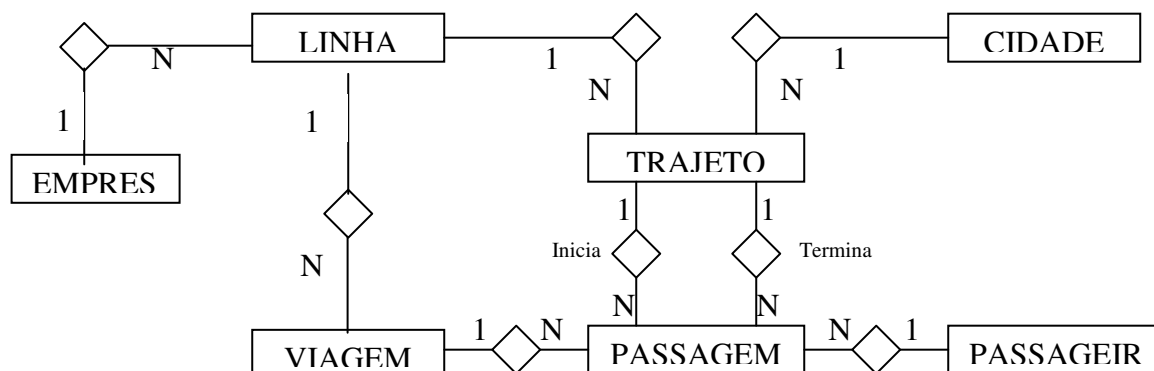
DETALHES DE PEDIDO	Colunas			
Linhas	NúmPedido	CodProduto	Quantidade	Desconto
01	0001	P01	10,00	20,20
02	0001	P02	15,00	
03	0002	P01	12,00	12,00
04	0003	P03	40,00	
05	0003	P04	20,00	5,40
06	0003	P05		
07	0003	P10	10,00	

PRODUTO	Colunas		
Linhas	Código	Nome	Preço
01	P01	Detergente y	0,50
02	P02	Desinfetante Z	1,80
03	P03	Açúcar 5 Kg	3,00
04	P04	Q-Boa	0,40
05	P05	Sabonete z	0,41
06	P01	Sabão em pó z	2,40
07	P06	Palha de aço K	0,20
08	P07	Arroz tipo z	5,40
09	P08	Feijão C	1,10

23. A matriz abaixo mostra comandos CREATE TABLE hipotéticos, aplicados nas tabelas do banco de dados relacional definidos na questão 21. Cada linha da matriz contém uma sequência de comandos *create table* que são executados da esquerda para a direita conforme indicado no título das colunas. Primeiro é executado o comando da coluna *Primeiro* da primeira linha, depois o da coluna *Segundo* da mesma linha e assim sucessivamente até a coluna *Quarto*. Quais tabelas são criadas como resultado da execução da sequência de comandos de cada linha da matriz?. Justificar a sua resposta.

Linhas	Primeiro	Segundo	Terceiro	Quarto
01	CREATE TABLE <i>Fornecedor ...</i>	CREATE TABLE <i>Produto ...</i>	CEATE TABLE <i>DetalhesDePedido...</i>	CREATE TABLE <i>Pedido ...</i>
02	CREATE TABLE <i>Pedido ...</i>	CREATE TABLE <i>Fornecedor ...</i>	CREATE TABLE <i>Produto ...</i>	CREATE TABLE <i>DetalhesDePedido...</i>
03	CREATE TABLE <i>Produto ...</i>	CREATE TABLE <i>Pedido ...</i>	CREATE TABLE <i>DetalhesDePedido...</i>	CREATE TABLE <i>Fornecedor ...</i>
04	CREATE TABLE <i>DetalhesDePedido...</i>	CREATE TABLE <i>Produto ...</i>	CEATE TABLE <i>Fornecedor ...</i>	CREATE TABLE <i>Pedido ...</i>
05	CREATE TABLE <i>Produto ...</i>	CREATE TABLE <i>Fornecedor ...</i>	CEATE TABLE <i>DetalhesDePedido...</i>	CREATE TABLE <i>Pedido ...</i>
06	CREATE TABLE <i>Produto ...</i>	CEATE TABLE <i>DetalhesDePedido...</i>	CREATE TABLE <i>Pedido ...</i>	CREATE TABLE <i>Fornecedor ...</i>
07	CREATE TABLE <i>Fornecedor ...</i>	CREATE TABLE <i>Produto ...</i>	CREATE TABLE <i>Pedido ...</i>	CEATE TABLE <i>DetalhesDePedido...</i>

24. O Banco de Dados Relacional definido, abaixo, refere-se a um ambiente de transporte interurbano de passageiros, via ônibus. Com base nesta definição resolver as três questões seguintes:



A entidade trajeto indica as cidades pelas quais passam as linhas de ônibus.

Definição das entidades e atributos:

EMPRESA = (Código, Nome)

LINHA = (Numero, CodigoEmpresa, DataDaConcessão)

TRAJETO = (Numero, NúmeroLinha, CodigoCidade, Ordem, HorárioDeChegada, HorárioDePartida)

CIDADE = (Código, Uf, Nome)

VIAGEM = (NumeroLinha, Numero, Data)

PASSAGEM = (NumeroLinha, NumeroViagem, Numero, NumeroPassageiro, NumeroTrajetoInicia, NumeroTrajetoTermina, Valor)

PASSAGEIRO = (Numero, Identidade, Sexo, Nome)

Observações:

- Legendas: Os atributos sublinhados são chaves nas tabelas onde aparecem
- Chaves Estrangeiras:
  - Na entidade LINHA: CódigoEmpresa relativa ao relacionamento entre MPRESA e LINHA.
  - Na entidade TRAJETO: NumeroLinha relativa ao relacionamento entre LINHA e TRAJETO e CodigoCidade relativa ao relacionamento entre CIDADE e TRAJETO.
  - Na entidade VIAGEM: NumeroLinha relativa ao relacionamento entre LINHA e VIAGEM.
  - Na entidade PASSAGEM: NumeroLinha e NumeroViagem relativa ao relacionamento entre VIAGEM e PASSAGEM; NumeroTrajetoInicia relativa ao relacionamento *Inicia* entre TRAJETO e PASSAGEM. NumeroTrajetoTermina relativa ao relacionamento *Termina* entre TRAJETO e PASSAGEM; NumeroPassageiro relativa ao relacionamento entre PASSAGEIRO e PASSAGEM.

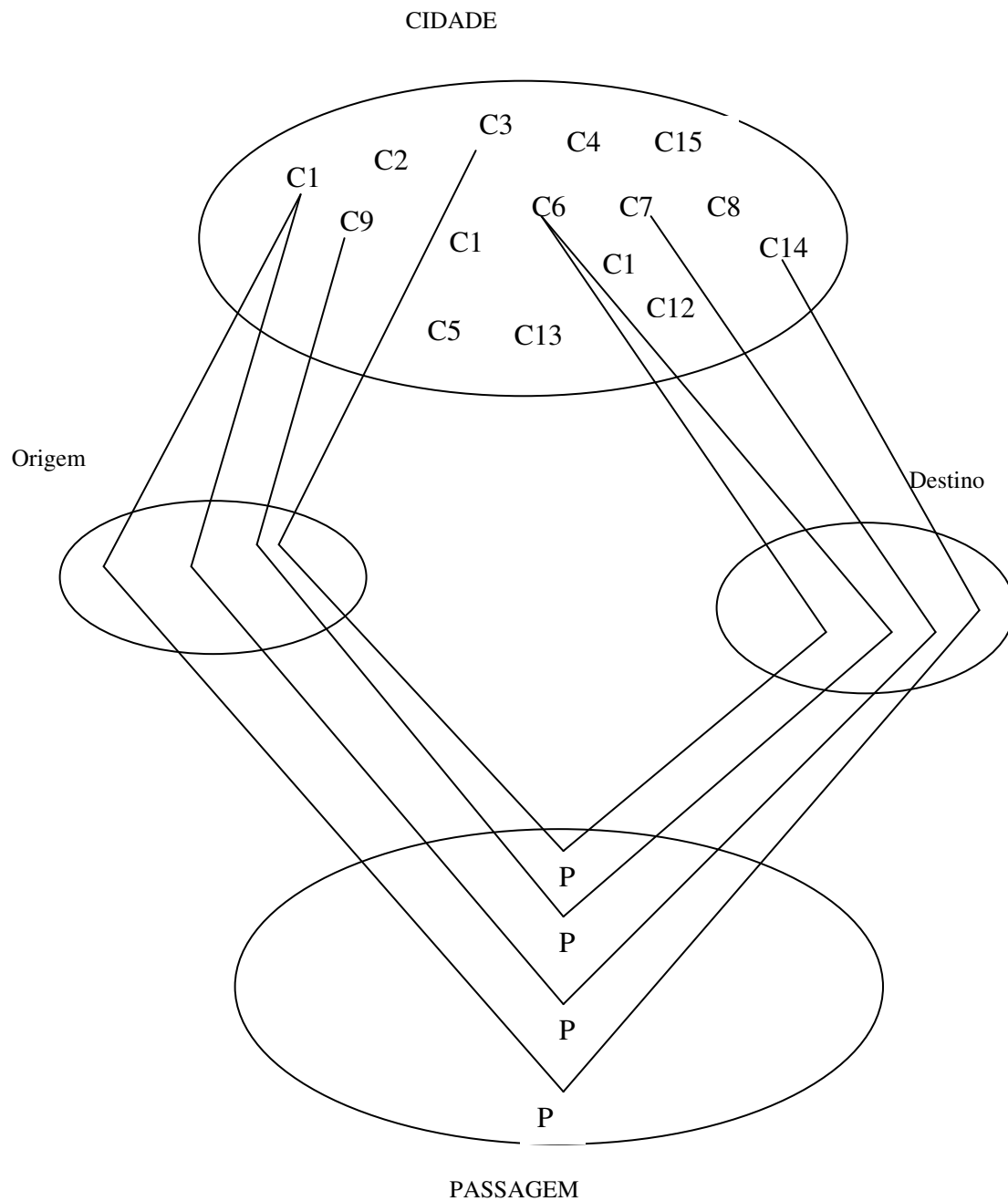


25. Listar as cidades correspondentes ao trajeto de cada linha de cada empresa. Mostrar na lista o nome da empresa, o número da linha e o código, uf e nome de cada cidade do trajeto da linha. Ordenar a lista por nome da empresa e número da linha.
26. Listar a quantidade de viagens feitas por linha e por empresa no ano de 2000. A lista deve conter o código e o nome da empresa, o número da linha e a quantidade de viagens feitas. A coluna correspondente à quantidade de viagens feitas deve ter o título: Viagens.
27. Listar o valor médio das passagens de cada viagem, por linha e empresa no primeiro semestre de 2001. Mostrar na lista: O nome da Empresa responsável pela linha, o número da linha, o número da viagem e o valor médio apurado. Ordenar a lista em ordem decrescente do valor médio apurado.
28. A Confederação Brasileira de Futebol – CBF vai promover um torneio de futebol entre os clubes: Flamengo, Goiás, Internacional de Porto Alegre e o São Paulo Futebol Clube, para homenagear o atleta do século – PELÉ. O torneio levará o verdadeiro nome do referido atleta e tem a seguinte tabela de realização dos jogos, mostrada abaixo. Construir um possível Diagrama de Entidade x Relacionamento – DER que possa dar origem à tabela mostrada.

Tabela dos jogos do torneio **Edson Arantes do Nascimento**

Turno	Rodada	Mandante	Visitante
Primeiro	01/07/2000	Flamengo	Goiás
		São Paulo	Internacional
	04/07/2000	São Paulo	Flamengo
		Internacional	Goiás
	08/07/2000	Internacional	Flamengo
		Goiás	São Paulo
Segundo	11/07/2000	Goiás	Flamengo
		Internacional	São Paulo
	14/07/2000	Flamengo	São Paulo
		Goiás	Internacional
	18/07/2000	Flamengo	Internacional
		São Paulo	Goiás

29. A figura abaixo mostra a correspondência entre as entidades CIDADE e PASSAGEM de um modelo de dados hipotético. Construir o DER correspondente as entidades descritas. Os atributos das entidades não precisam ser mostrados..



30. Construir o DER, completo, inclusive mostrando as entidades fracas, correspondente às relações definidas abaixo:

<b>PESSOA</b>	Atributos	Identificador	Rel	Tipo	Tamanho	Requerido
	Codigo	S		Number	4	S
	Nome			Text	30	S
	DataNascimento			Date		S
	Sexo			Text	1	S
	NumeroCasamento		S			S

<b>CASAMENTO</b>	Atributos	Identificador	Rel	Tipo	Tamanho	Requerido
	Numero	S		Number	8	S
	CodigoEsposo		S			S
	CodigoEsposa		S			S
	DataCasamento			Date		S

<b>TORCEDOR</b>	Atributos	Identificador	Rel	Tipo	Tamanho	Requerido
	CodigoPessoa	S	S			S
	CodigoClube	S	S			S

<b>CLUBE</b>	Atributos	Identificador	CE	Tipo	Tamanho	Requerido
	Codigo	S		Number	4	S
	Nome			Text	30	S
	Cidade			Text	30	S
	Estado			Text	2	S
	DataFundacao			Date		S

Obs:

- 1) O NumeroCasamento, na relação **PESSOA**, indica que a pessoa é filha das pessoas indicadas na relação **CASAMENTO**.
  - 2) Há pessoas que não são casadas.
  - 3) Há pessoas que não são torcedoras de nenhuma clube.
  - 4) Há pessoas que são casadas mais de uma vez e há pessoas com mais de um filho.
31. Que alteração deveríamos fazer no DER e nas relações apresentadas na questão anterior, para representar a situação de não se saber quem são os pais de uma determinada pessoa?
32. Ainda em relação ao DER da questão 30, listar as pessoas casadas mais de uma vez. A lista deve conter: O código, o nome e a quantidade de casamentos da pessoa.
33. Ainda em relação ao DER da questão 30, listar as pessoas que torcem para o FLAMENGO. A lista deve conter todos os dados da Pessoa e deve ser apresentada em ordem crescente de nome da pessoa.

34. Obter uma relação de passageiros, como a mostrada abaixo, para cada linha da empresa ARAGUAIA, do dia 10/10/2000 da linha 120.

Passageiro	Cidade Embarque	Uf	Cidade Desembarque	Uf	Valor
Carlos Figueiredo	Uberaba	Mg	Franca	Sp	50,00
Fernando Rassi	Morrinhos	Go	São Paulo	Sp	80,00
Francisco Beltrão	Goiânia	Go	Campínas	Sp	70,00
Luiz Ricardo	Goiânia	Go	Uberlândia	Mg	30,00
.....	.....	...	.....	...	.....
.	...	.	...	.	.
.....	.....	...	.....	...	.....
.	...	.	...	.	.
.....	.....	...	.....	...	.....
.	...	.	...	.	.
.....	.....	...	.....	...	.....
.	...	.	...	.	.
Maria dos Santos	Goiânia	Go	Ribeirão Preto	Sp	60,00
Maria Ribeiro	Goiânia	Go	Limeira	Sp	70,00
Sandro Rivera	Goiânia	Go	São Paulo	Sp	90,00

Observações:

- Observe que a relação deve ser mostrada em ordem alfabética por nome do passageiro.
  - As linhas pontilhadas representam dados hipotéticos.
  - Cidade Embarque é a cidade onde cada passagem se inicia.
  - Cidade Desembarque é a cidade onde a passagem termina.
35. Escrever, na ordem correta, os comandos CREATE's para criar as tabelas correspondentes as descrições dos atributos, abaixo:

ALMOXARIFADO	Atributos	Primary Key	Foreign Key	Tipo	Tamanho	Requerido
	Codigo	S		Number	4	S
	Nome			Text	30	S
	Endereço			Text	60	S

ESTOQUE	Atributos	Primary Key	Foreign Key	Tipo	Tamanho	Requerido
	CodigoAlmoxarifado	S	S	Number	4	S
	CodigoProduto	S	S	Number	4	S
	Saldo			Number	7	

PRODUTO	Atributos	Primary Key	Foreign Key	Tipo	Tamanho	Requerido
	Código	S		Text	3	S
	Nome			Text	30	S
	EstoqueMinimo			Number	7	S

Obs:

- 1) Uma ocorrência de ALMOXARIFADO pode ter uma ou mais ocorrências de ESTOQUE a ela vinculada.
- 2) Uma ocorrência de ESTOQUE deve se vincular a uma e somente uma ocorrência de ALMOXARIFADO.
- 3) Uma ocorrência de PRODUTO pode ter uma ou mais ocorrências de ESTOQUE a ela vinculada
- 4) Uma ocorrência de ESTOQUE deve ser vinculada a uma e somente uma ocorrência de PRODUTO.
- 5) O atributo CódigoAlmoxarifado é Foreign Key de ALMOXARIFADO em ESTOQUE.
- 6) O atributo CódigoProduto é a Foreign Key de PRODUTO EM ESTOQUE.

36. Escrever os comandos INSERT INTO, na ordem correta, para incluir os dados relacionados, abaixo:

ALMOXARIFADO	Código	Nome	Endereço
	0001	São Jorge	Rua 15 Centro
	0002	Vaca Brava	Rua T-10

ESTOQUE	CódigoAlmoxarifado	CódigoProduto	Saldo
	0001	P01	15
	0002	P02	18

PRODUTO	Código	Nome	EstoqueMínimo
	P01	Sofá	10
	P02	Mesa de Mármore	20

37. Tomando como base as relações da questão 36: Listar os saldos de cada produto por almoxarifado. A lista deve conter todos os atributos de almoxarifado, todos de produto e o saldo em estoque do produto no estoque. Mostrar a lista ordenada por nome do produto.
38. Tomando como base as relações da questão 36: Listar os produtos cujos saldos em estoque sejam menor ou igual ao EstoqueMínimo do produto. Mostrar na lista o código, nome e estoque mínimo do produto e o saldo do produto em estoque. Mostrar a lista ordenada por nome do produto.
39. Tomando como base as relações da questão 36: Listar o total do saldo de cada produto em estoque. A lista deve conter: O código e o nome do produto e o total do saldo em estoque. A coluna correspondente ao total do saldo deve ter o título: Saldo.
40. Tomando como base as relações da questão 36: Escreva um comando INSERT INTO, para a tabela PRODUTO, que viole as restrições de integridade de chave.
41. Tomando como base as relações da questão 36: Escreva um comando DELETE, para a tabela ALMOXARIFADO, que viole as restrições de integridade referencial.
42. Tomando como base as relações da questão 36: Listar os produtos que não tenham nenhuma ocorrência na tabela ESTOQUE. A lista deve conter todos os atributos de PRODUTO e deve ser mostrada em ordem crescente de código do produto.
43. Tomando como base as relações da questão 36: Criar uma cópia da tabela ESTOQUE com os mesmos atributos e o mesmo conteúdo da tabela original.
44. Tomando como base as relações da questão 36: Alterar a tabela PRODUTO para que passe a conter o atributo Unidade, com três caracteres de tamanho.

45. Colocar os comandos abaixo, na ordem correta para que as tabelas possam ser criadas e construir o DER correspondente.

Comando 1:

```
CREATE TABLE Concedente
  (Codigo NUMBER (3) NOT NULL,
   Nome CHAR (30) NOT NULL,
   CONSTRAINT Concedente PRIMARY KEY (Codigo))
```

Comando 2:

```
CREATE TABLE PromocaoProduto
  (Codigo NUMBER (3) NOT NULL,
   Numero NUMBER (3) NOT NULL,
   Produto NUMBER (3) NOT NULL,
   CONSTRAINT PromocaoProduto PRIMARY KEY (Codigo, Numero, Produto),
   CONSTRAINT PromocaoProduto1 FOREIGN KEY (Codigo, Numero)
     REFERENCES Promocao (Codigo,Numero),
   CONSTRAINT PromocaoProduto2 FOREIGN KEY (Produto)
     REFERENCES Produto (Codigo))
```

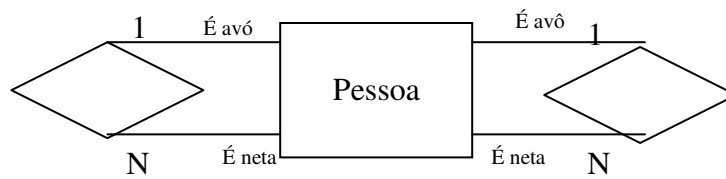
Comando 3:

```
CREATE TABLE Promocao
  (Codigo NUMBER (3) NOT NULL,
   Numero NUMBER (3) NOT NULL,
   Categoria CHAR (1) NOT NULL,
   Objeto CHAR (1) NOT NULL,
   Beneficiario CHAR (1) NOT NULL,
   CONSTRAINT Promocao PRIMARY KEY (Codigo, Numero),
   CONSTRAINT Promocao1 FOREIGN KEY (Codigo)
     REFERENCES Concedente (Codigo))
```

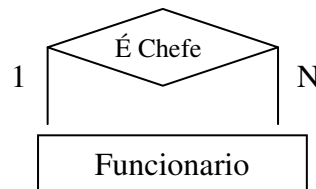
Comando 4:

```
CREATE TABLE Produto
  (Codigo NUMBER (3) NOT NULL,
   Nome CHAR (30) NOT NULL,
   Valor NUMBER (8),
   CONSTRAINT Produto PRIMARY KEY (Codigo))
```

46. Escreva o comando CREATE TABLE, para criar a balela e os relacionamentos representativos do DER abaixo. Sugerir, pelo menos, quatro para a tabela criada.



47. Escreva um comando, CREATE TABLE, para criar a tabela e o relacionamento representativos do DER abaixo. Sugerir, pelo menos, quatro atributos para a tabela criada.



48. Um comando DELETE sempre viola uma restrição ou restrições de integridade? Justifique sua resposta.

## Bibliografia

### Bibliografia

1. Rumbaugh, James e outros. **Modelagem e Projetos Baseados em Objetos**. Rio de Janeiro: Campus, 1988.
2. Pompilho, S. **Análise Essencial**. Rio de Janeiro: Infobook, 1995.
3. Elmasri, Ramez e Navathe, Shamkant B.  
    **“Fundamentals of Database Systems”**,  
    Benjamim/Cummings Publishing Company, U.S.A, 2ª Edition
4. Korth, Henry e Silberschatz, Abraham – (*Livro Texto Básico*)  
    **“Sistemas de Bancos de Dados”**,  
    Makron Books do Brasil Editora Ltda, RJ, 1994, 2ª Edição
5. RUMBAUGH, JAMES e outros. **Modelagem e Projetos Baseados em Objetos**. Rio de Janeiro: Campus, 1988.
6. Oracle. **Data Modelling and Database Design**. California. Oracle Corporation, 1992.