

# Introdução a Teoria de Bancos de Dados Relacional e SQL

# Banco de Dados Relacional e SQL

- Servidor de Banco de Dados vs Arquivos
- Introdução à teoria de Banco de Dados
- Diagrama Entidade Relacionamento
- Tipos de dados em Banco de Dados
- Introdução a SQL

## Sugestão de Bibliografia:

- *SQL Curso Prático*  
Oliveira, Celso Henrique
- *Relational Database Design Clearly Explained*  
Harrington, Jan

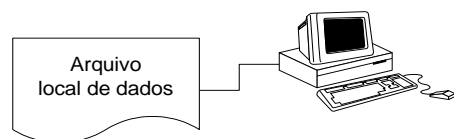
# Servidores de Banco de Dados vs Arquivos

Principais formas de armazenamento permanente de dados:

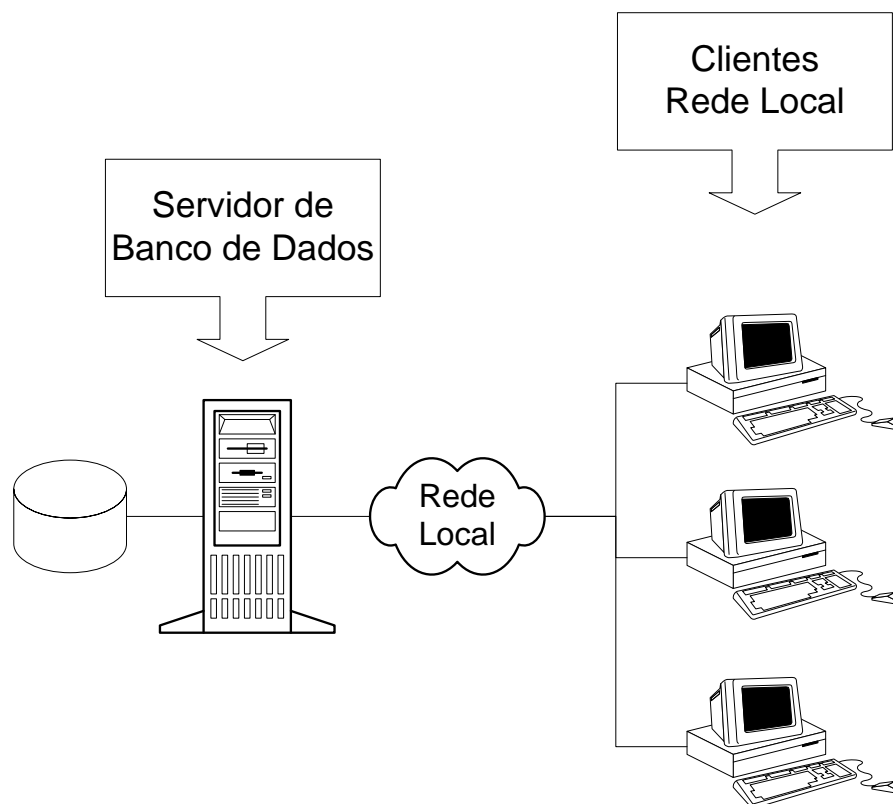
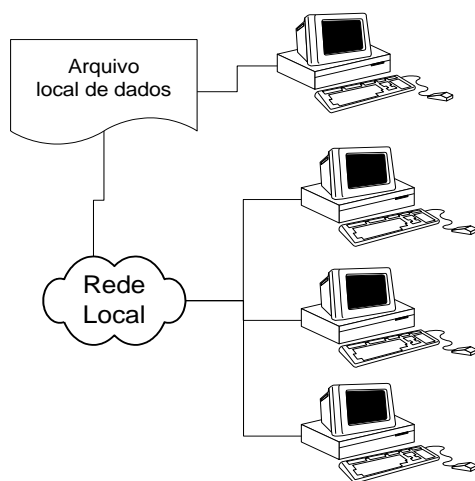
1. **Arquivos:** arquivos texto, planilhas eletrônicas, arquivos em formatos especiais como por exemplo MS-Access (.mdb).

## 2. Servidor de Banco de Dados

Computador isolado



Computadores em rede

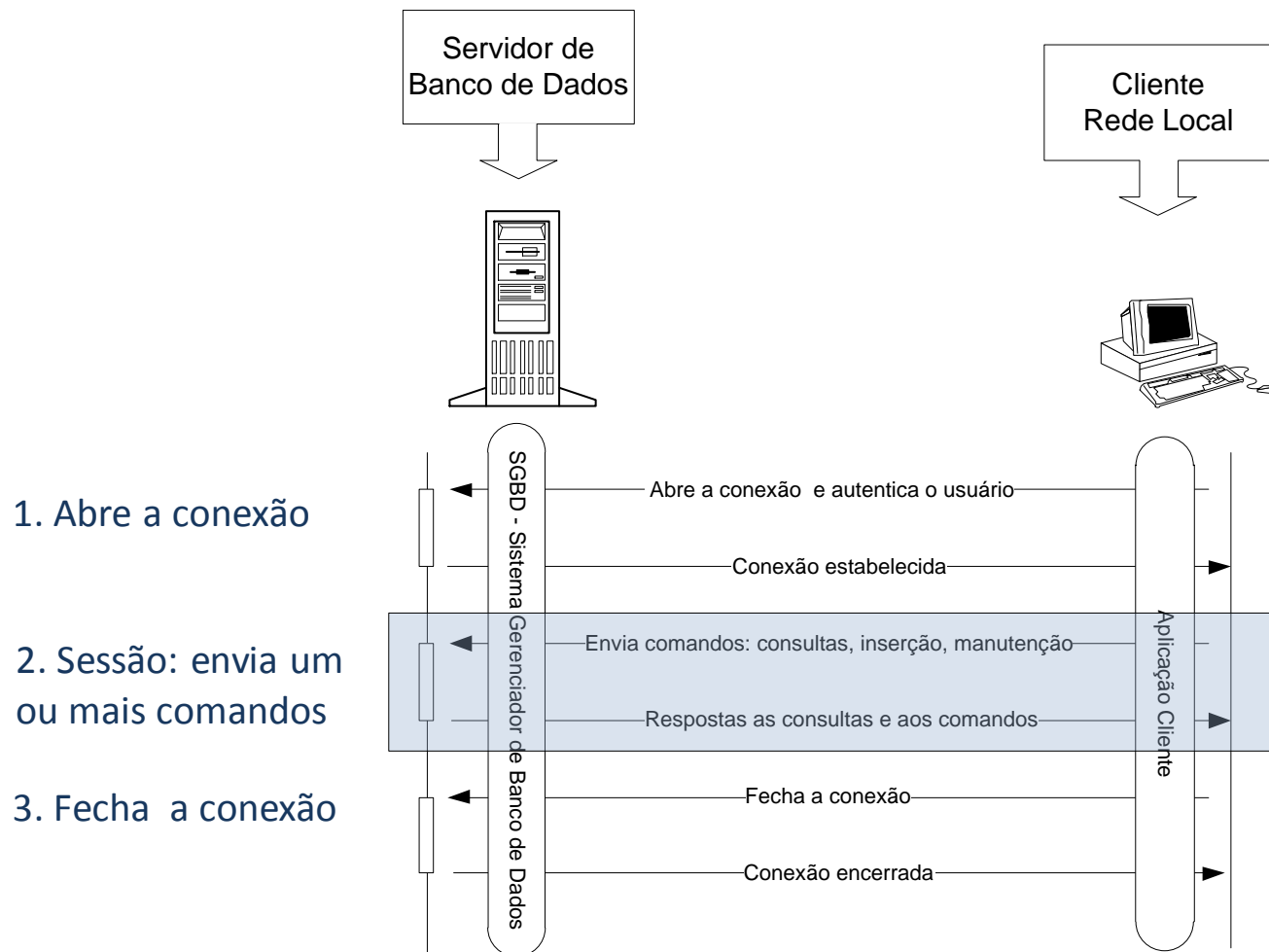


# Servidores de Banco de Dados vs Arquivos

Vantagens de se utilizar um Servidor de Banco de Dados:

- **Garante a consistência do conteúdo e do relacionamento entre os dados**
- **Grande capacidade de armazenamento e velocidade para recuperação dos dados – consultas não sequenciais**
- Possibilita o acesso de vários usuários simultaneamente para leitura e escrita
- Controle de Acesso: Restringe o acesso para consultar/escrever/remover registros por tabela e campo
- Registra os comandos e as operações executados pelos usuários
- Suporta a consultas sofisticadas relacionando e agrupando dados
- Suporte a procedimentos Transacionais para garantir a integridade dos dados
- Gerenciamento do acesso dos usuários ao banco de dados
- Centraliza o gerenciamento dos dados (backup/atualização)
- Automatização de Tarefas: backup, *Stored Procedures* e *Triggers*

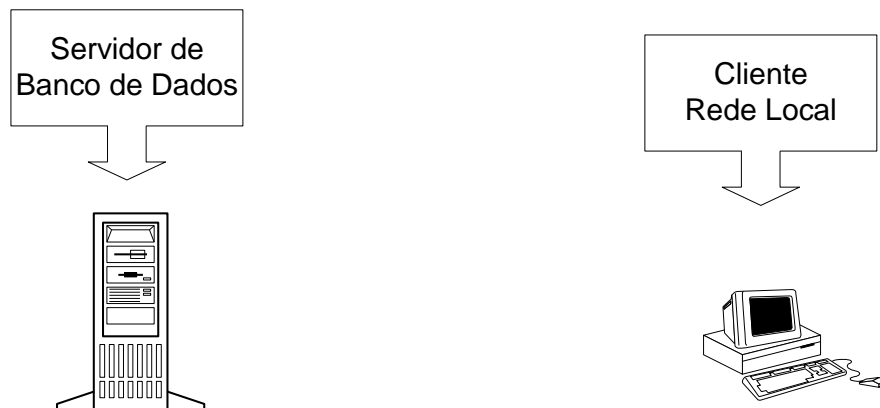
# Acesso a Servidores de Banco de Dados



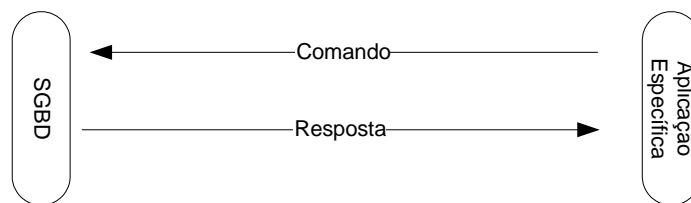
- Gratuitos:
  - MySQL
  - Postgress
  - Firebird
- Comerciais:
  - Oracle
  - Sysbase
  - SQL Server
  - “MS Access”

# Acesso a Servidores de Banco de Dados

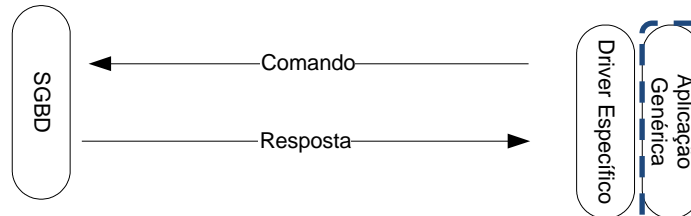
## Alternativas para conexão com Servidores de Banco de Dados



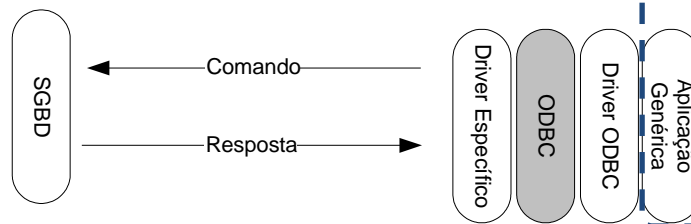
### Aplicação Específica:



### Driver Específico:



### Driver ODBC: (MS Windows)



R, Matlab,  
Excel, Etc..

# Introdução a Teoria de Banco de Dados

Em um Banco de Dados Relacional os dados e o relacionamento entre os dados estão organizados em: Tabelas, Colunas e Linhas

Exemplo de um banco de dados simples com uma única tabela:

ID	ISBN	Título	Área	Editora	Autores
1	0262022834	LECTURES ON MACROECONOMICS	macroeconomia	MIT PRESS	Stanley Fisher, Oliver Blanchard
2	8535208798	MACROECONOMIA	macroeconomia	CAMPUS	Oliver Blanchard
3	8587918443	MACROECONOMIA	macroeconomia	PRENTICE HALL BRASIL	Oliver Blanchard
4	0262521814	REFORM IN EASTERN EUROPE	macroeconomia	MIT PRESS	Oliver Blanchard, Rudiger Dornbush, Paul Krugma
5	8522103712	PRINCIPIOS DE MICROECONOMIA	microeconomia	THOMSON PIONEIRA	Gregory Mankiw
6	853521044X	INTRODUÇÃO À MICROECONOMIA	microeconomia	CAMPUS	Joseph Stiglitz
7	8576050188	MICROECONOMIA	microeconomia	PRENTICE HALL BRASIL	Daniel Rubinfeld, Robert Pindyck
8	8535216707	MICROECONOMIA - PRINCÍPIOS BÁSICOS	microeconomia	CAMPUS	Hal Variant
9	8522104212	INTRODUÇÃO À ECONOMETRIA	econometria	THOMSON PIONEIRA	Jeffrey Woodridge
10	0324113641	INTRODUCTORY ECONOMETRICS	econometria	IE-THOMSON	Jeffrey Woodridge
11	8535213430	ECONOMETRIA - MODELOS E PREVISÕES	econometria	CAMPUS	Daniel Rubinfeld, Robert Pindyck
12	8502041231	ESTATÍSTICA E INTRODUÇÃO À ECONOMETRIA	econometria	SARAIVA	Alexandre Sartóris

# Introdução a Teoria de Banco de Dados

## Banco de Dados Relacional:

- **TABELA:** Estrutura primitiva básica
  - Uma tabela esta associada a uma **Entidade**.
  - Entidade: “qualquer coisa” tangível ou intangível (associação ou relacionamento)
- **COLUNAS:** Atributos ou Campos da entidade
- **LINHAS:** Registros de valores para dos atributos que descrevem uma única ocorrência da entidade no mundo real: uma *instância* da entidade

As entidades estão relacionadas entre si de forma **consistente e lógica**, compondo um modelo cujo objetivo principal é representar algum aspecto do mundo real.



# Introdução a Teoria de Banco de Dados

## Exemplo de Entidade, Atributos e Registros

### Tabela/Entidade: Livro

**Atributos:** ID, ISBN, Título, Área, Editora, Autores, Ano, Edição e Idioma....

### Registro:

ID	ISBN	Título	Área	Editora	Autores
1	0262022834	LECTURES ON MACROECONOMICS	macroeconomia	MIT PRESS	Stanley Fisher, Oliver Blanchard
2	8535208798	MACROECONOMIA	macroeconomia	CAMPUS	Oliver Blanchard
3	8587918443	MACROECONOMIA	macroeconomia	PRENTICE HALL BRASIL	Oliver Blanchard
4	0262521814	REFORM IN EASTERN EUROPE	macroeconomia	MIT PRESS	Oliver Blanchard, Rudiger Dornbush, Paul Krugman
5	8522103712	PRINCIPIOS DE MICROECONOMIA	microeconomia	THOMSON PIONEIRA	Gregory Mankiw
6	853521044X	INTRODUÇÃO À MICROECONOMIA	microeconomia	CAMPUS	Joseph Stiglitz
7	8576050188	MICROECONOMIA	microeconomia	PRENTICE HALL BRASIL	Daniel Rubinfeld, Robert Pindyck
8	8535216707	MICROECONOMIA - PRINCÍPIOS BÁSICOS	microeconomia	CAMPUS	Hal Variant
9	8522104212	INTRODUÇÃO À ECONOMETRIA	econometria	THOMSON PIONEIRA	Jeffrey Woodridge
10	0324113641	INTRODUCTORY ECONOMETRICS	econometria	IE-THOMSON	Jeffrey Woodridge
11	8535213430	ECONOMETRIA - MODELOS E PREVISÕES	econometria	CAMPUS	Daniel Rubinfeld, Robert Pindyck
12	8502041231	ESTATÍSTICA E INTRODUÇÃO À ECONOMETRIA	econometria	SARAIVA	Alexandre Sartóris

# Tipos de Relacionamento

ID	ISBN	Título	Área	Editora	Autores
1	0262022834	LECTURES ON MACROECONOMICS	macroeconomia	MIT PRESS	Stanley Fisher, Oliver Blanchard
2	8535208798	MACROECONOMIA	macroeconomia	CAMPUS	Oliver Blanchard
3	8587918443	MACROECONOMIA	macroeconomia	PRENTICE HALL BRASIL	Oliver Blanchard
4	0262521814	REFORM IN EASTERN EUROPE	macroeconomia	MIT PRESS	Oliver Blanchard, Rudiger Dornbush, Paul Krugma
5	8522103712	PRINCIPIOS DE MICROECONOMIA	microeconomia	THOMSON PIONEIRA	Gregory Mankiw
6	853521044X	INTRODUÇÃO À MICROECONOMIA	microeconomia	CAMPUS	Joseph Stiglitz
7	8576050188	MICROECONOMIA	microeconomia	PRENTICE HALL BRASIL	Daniel Rubinfeld, Robert Pindyck
8	8535216707	MICROECONOMIA - PRINCÍPIOS BÁSICOS	microeconomia	CAMPUS	Hal Variant
9	8522104212	INTRODUÇÃO À ECONOMETRIA	econometria	THOMSON PIONEIRA	Jeffrey Woodridge
10	0324113641	INTRODUCTORY ECONOMETRICS	econometria	IE-THOMSON	Jeffrey Woodridge
11	8535213430	ECONOMETRIA - MODELOS E PREVISÕES	econometria	CAMPUS	Daniel Rubinfeld, Robert Pindyck
12	8502041231	ESTADÍSTICA E INTRODUÇÃO À ECONOMETRIA	econometria	SARAIVA	Alexandre Sartoris

Relacionamento	Exemplo
1:1 Um-para-Um	Um livro tem um código ISBN Um código ISBN está relacionado a um único livro
1:N Um-para-N	Uma editora publica um ou mais livros Um livro foi publicado por uma única editora
N:N N-para-N	Um autor escreve 1 ou mais livros Um livro pode ser escrito por mais de um autor

# Tipos de Relacionamento

## Problemas:

- Duplicar as informações na base de dados
- Dificuldade de realizar a busca por falta de padronização e por estar em um único campo

0 Dificuldade de manutenção: Modificar o endereço da editora ou os dados de uma autor

## Solução:

- Representar cada entidade no banco de dados por uma tabela
- Identificar claramente o tipo de relacionamento entre as entidades

# Tabelas e Chaves

Uma base de dados deve conter:

- Os atributos as entidades
- As associações entre as entidades

Devemos criar:

- Criar uma tabela para cada entidade
- Utilizar *chaves* para identificar os registros das tabelas
- Criar tabelas de relacionamento entre entidades

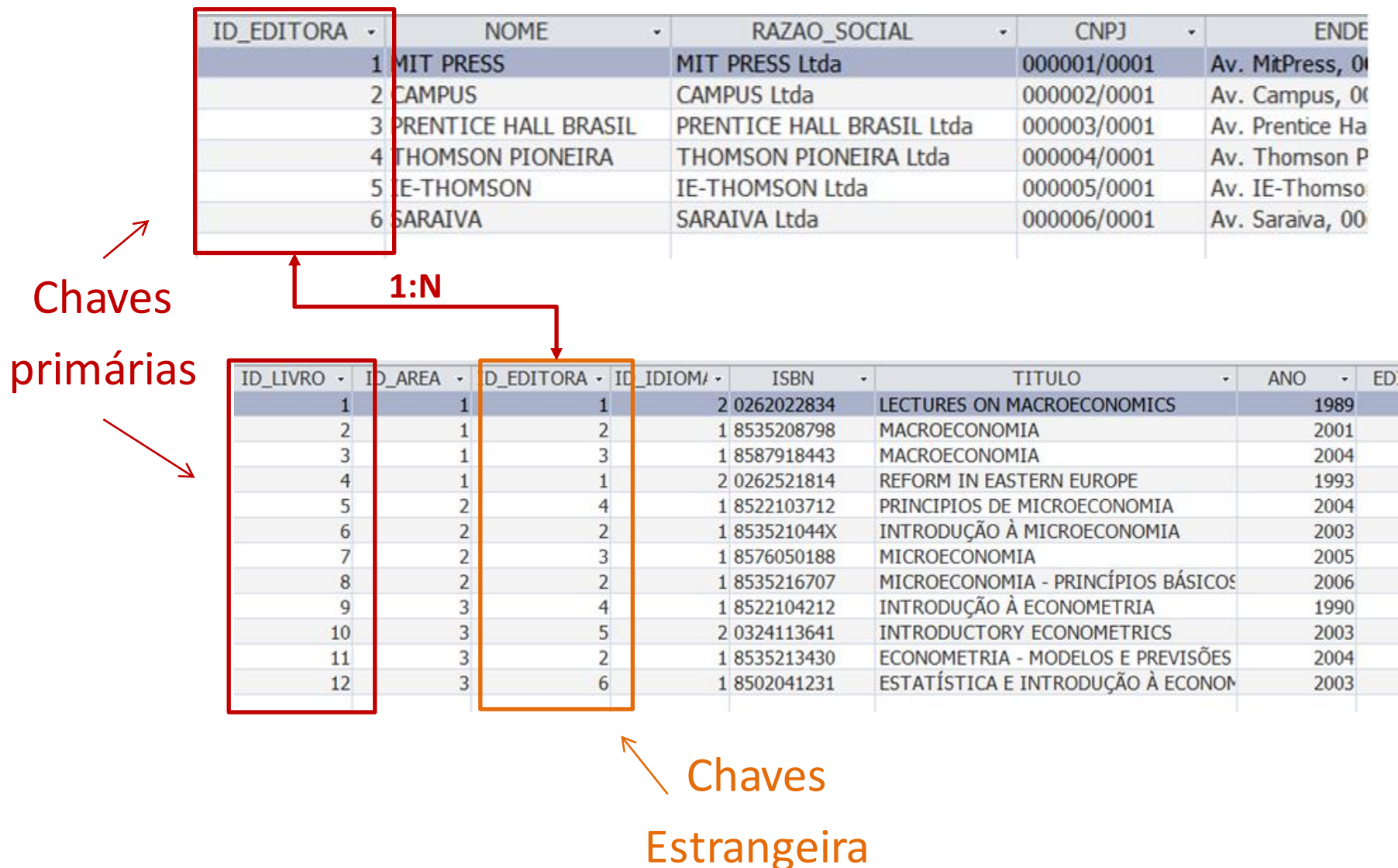
## Tabela: Editora

**chave**



ID_EDITORA	NOME	RAZAO_SOCIAL	CNPJ	ENDE
1	MIT PRESS	MIT PRESS Ltda	000001/0001	Av. MitPress, 00
2	CAMPUS	CAMPUS Ltda	000002/0001	Av. Campus, 00
3	PRENTICE HALL BRASIL	PRENTICE HALL BRASIL Ltda	000003/0001	Av. Prentice Ha
4	THOMSON PIONEIRA	THOMSON PIONEIRA Ltda	000004/0001	Av. Thomson P
5	IE-THOMSON	IE-THOMSON Ltda	000005/0001	Av. IE-Thomso
6	SARAIVA	SARAIVA Ltda	000006/0001	Av. Saraiva, 00

# Chave Primária e Chave Estrangeira



# Tipos de Chaves: Chaves Primárias

## Chaves Primárias

- Identifica de forma única uma instância da entidade.
- Atributo especial presente em **todas** as entidades, com valor **único, não nulo, exclusivo e imutável para cada linha**
- Boas Prática:

- Criar chaves primárias com a exclusiva função de ser chave primária, ou seja, não deve ter um significado intrínseco.

Ex: Ao criar uma tabela de Pessoas não utilizar CPF ou RG como chave primária.

- Normalmente a chave primária de uma tabela é um número natural e os bancos de dados incrementam o contador automaticamente à medida que novos registros são inseridos.

# Tipos de Chaves: Chaves Estrangeiras

## Chaves Estrangeiras

- Estabelece relações entre instancias de entidades distintas.
- O valor da chave estrangeira na instância de uma entidade, corresponde ao valor da chave primária da instância de uma entidade com a qual ela esta relacionada.

Exemplo:

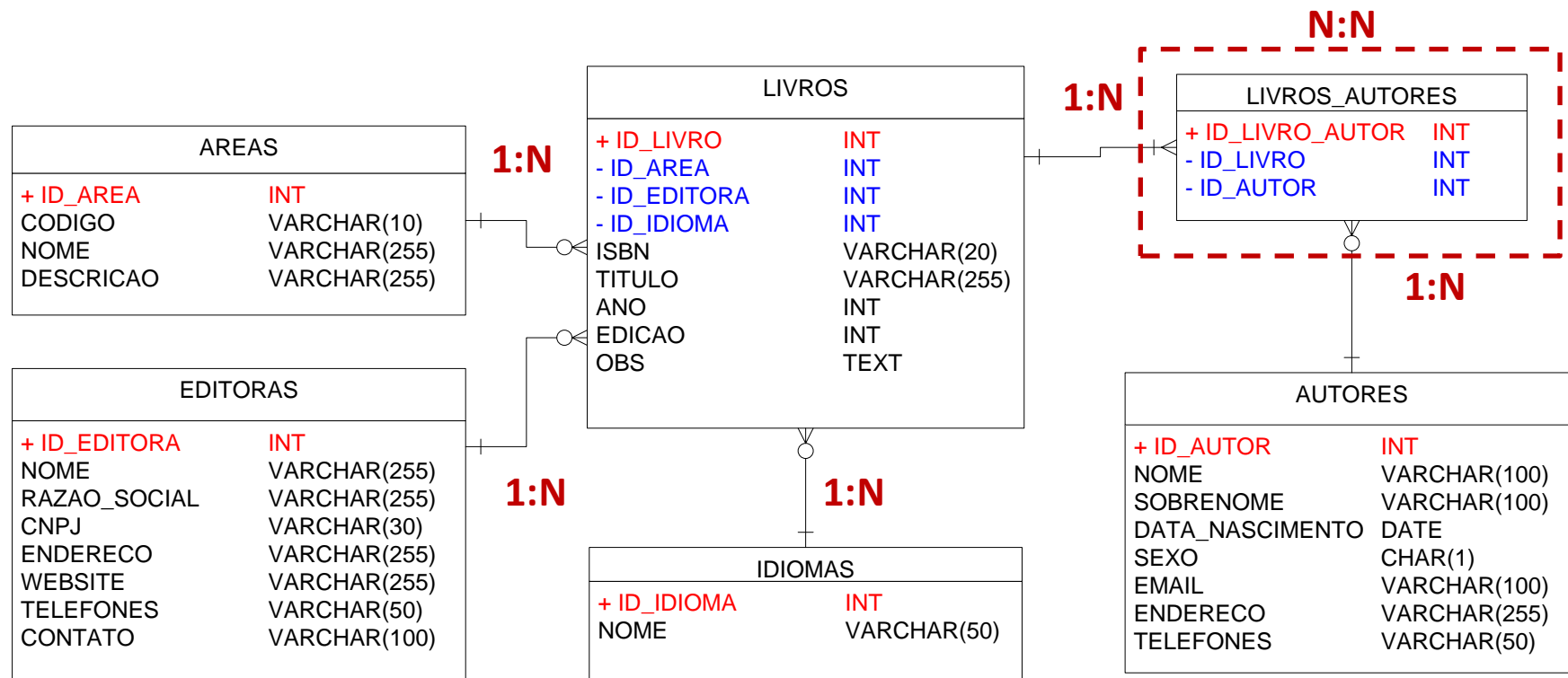
Relação: “a entidade Livro esta relacionada com a entidade Editora”.

Cada entidade Editora criamos uma chave primária ID\_EDITORA.

Quando inserimos um novo Livro na base de dados, ao invés de copiar todos os atributos da Editora, apenas inserimos no registro que define a nova instancia do Livro a chave primária que identifica a Editora.

- Pode ter valor nulo quando o relacionamento não for obrigatório.
- Uma entidade pode conter uma ou mais chaves estrangeiras dependendo do relacionamento que esta entidade estabelece com as demais.

Representação das entidades e dos relacionamentos entre as mesmas em um banco de dados.





## 3 Regras Básicas:

1. Cada entidade no Diagrama ER será representada por uma caixa que deverá conter:

- O nome da entidade, no plural ou singular dependendo da preferência do autor.
- A lista de atributos da entidade, incluindo chaves primárias e estrangeiras.
- Opcionalmente pode-se listar o tipo de cada atributo ou utilizar símbolos para indicar unicidade ou obrigatoriedade do campo.

Exemplo de Convenção:

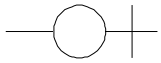

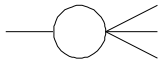
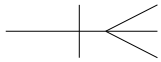
‘+’, ou escrever PK (primary key)

‘-’ ou escrever FK (foreign key)

# Diagrama Entidade Relacionamento

## 3 Regras Básicas (continuação...):

2. Tabelas relacionadas diretamente entre si são unidas por linhas. As terminações da linha indicam o tipo de relacionamento. Quatro terminações são possíveis:

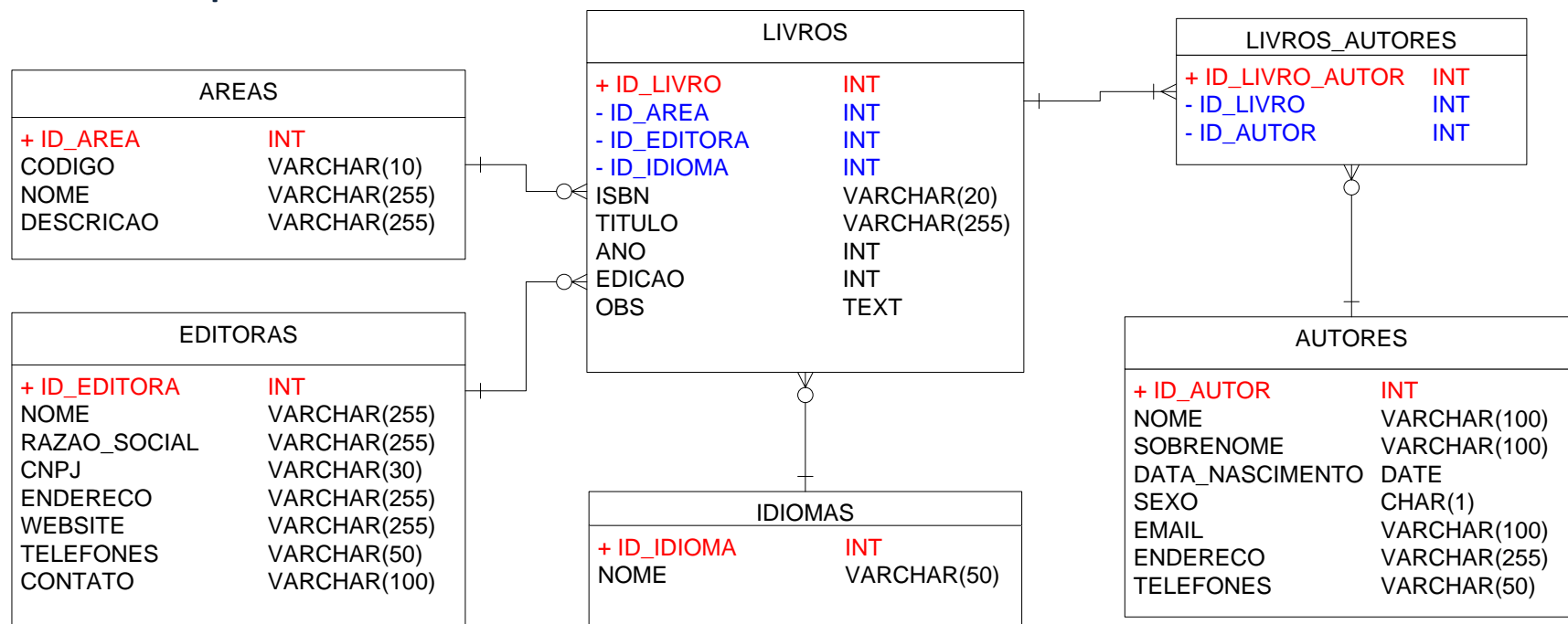
Terminação	Significado
	Zero ou uma
	Obrigatoriamente uma e apenas uma
	Zero, uma ou várias
	Obrigatoriamente uma ou várias

3. Por convenção recomenda-se que as Chaves Primárias mantenham o mesmo nome quando forem utilizadas como Chaves Estrangeiras em outras entidades.

# Diagrama Entidade Relacionamento

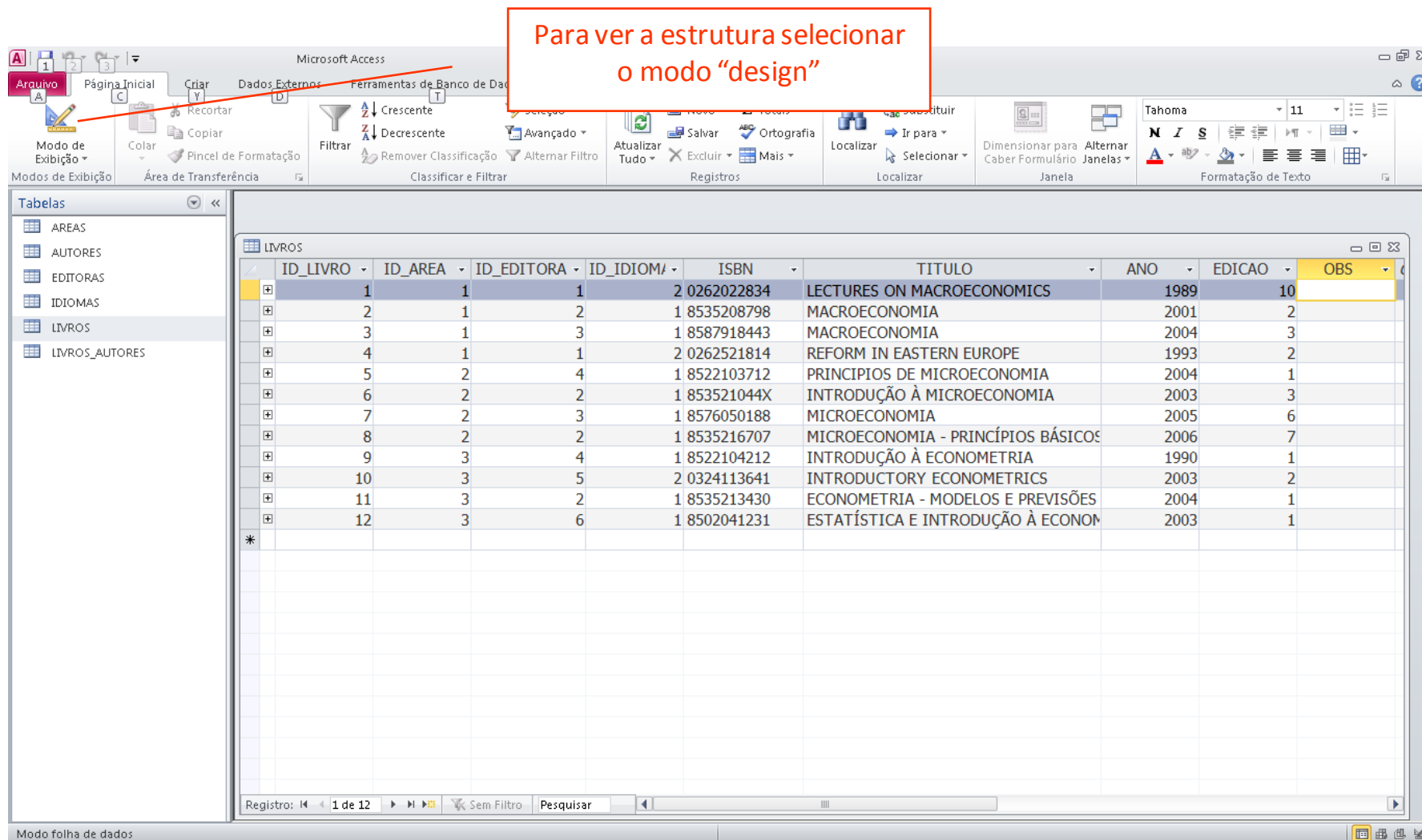
Alguns relacionamentos:

- um editora pode publicar nenhum, um ou mais livros
- um autor pode escrever nenhum, um ou mais livros
- um livro necessariamente foi publicado por uma editora
- um livro pode ter um ou mais autores



Abrir a bse de dados: *livros.bd1.mdb* utilizando o *MS-Access*

Para ver a estrutura selecionar o modo "design"

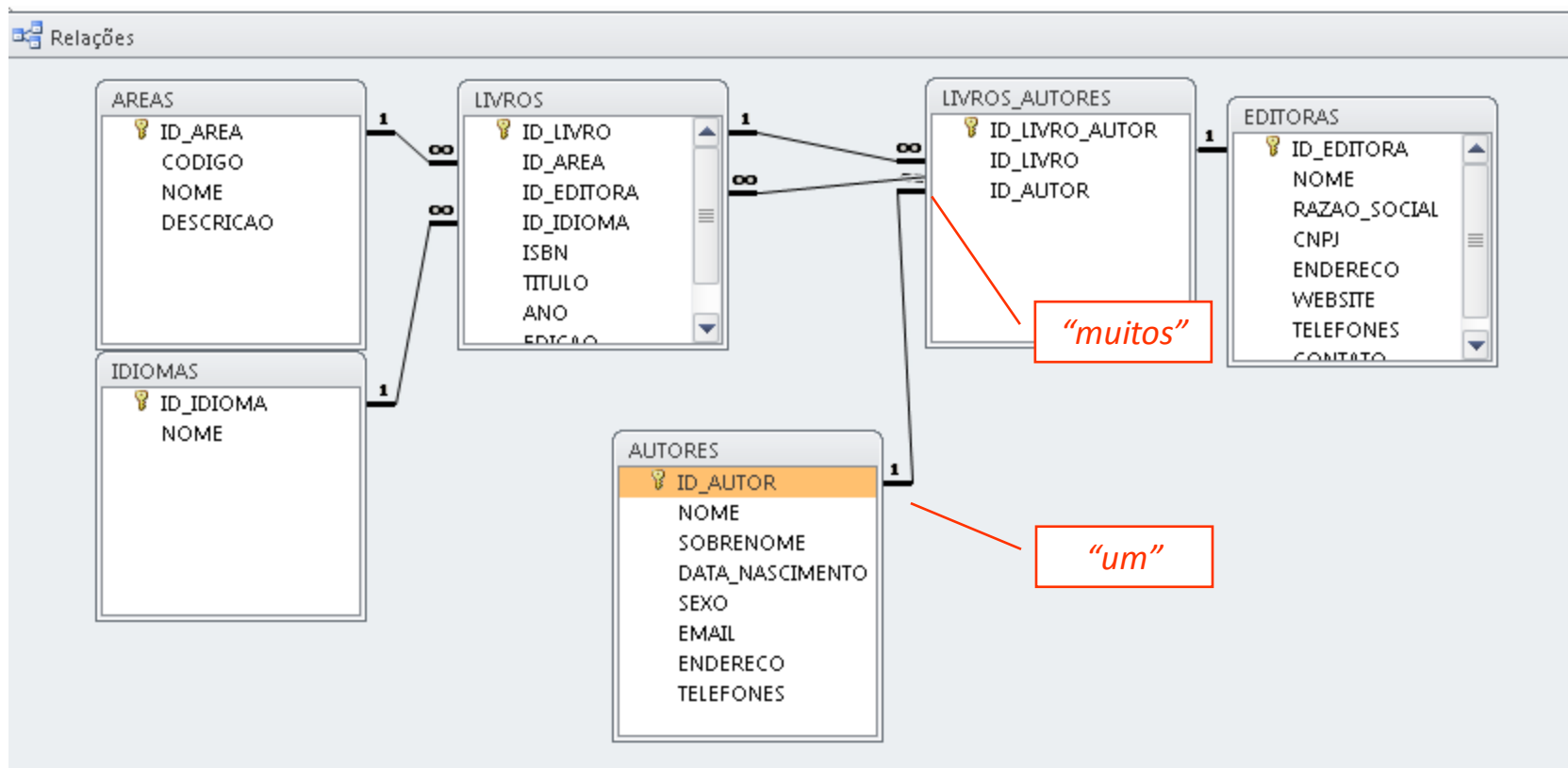
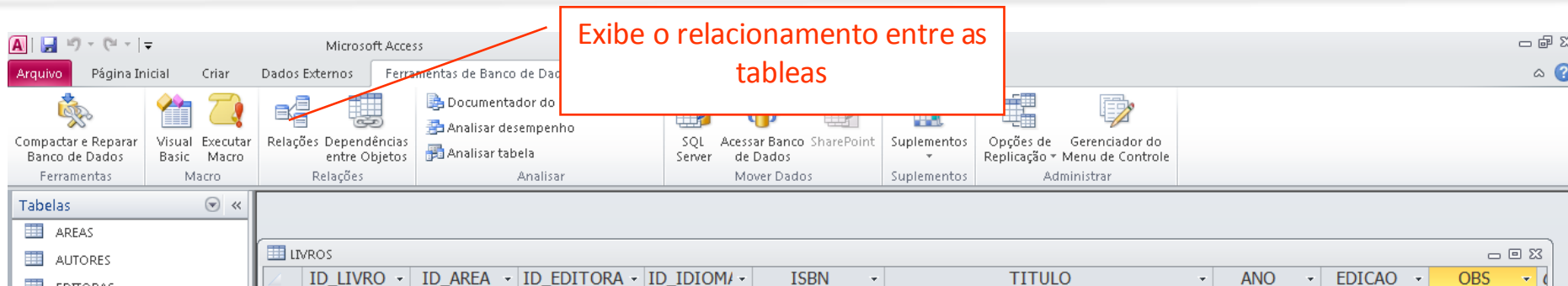


Microsoft Access - Tabela: LIVROS

ID_LIVRO	ID_AREA	ID_EDITORA	ID_IDIOM	ISBN	TITULO	ANO	EDICAO	OBS
1	1	1	2	0262022834	LECTURES ON MACROECONOMICS	1989	10	
2	1	2	1	8535208798	MACROECONOMIA	2001	2	
3	1	3	1	8587918443	MACROECONOMIA	2004	3	
4	1	1	2	0262521814	REFORM IN EASTERN EUROPE	1993	2	
5	2	4	1	8522103712	PRINCIPIOS DE MICROECONOMIA	2004	1	
6	2	2	1	853521044X	INTRODUÇÃO À MICROECONOMIA	2003	3	
7	2	3	1	8576050188	MICROECONOMIA	2005	6	
8	2	2	1	8535216707	MICROECONOMIA - PRINCÍPIOS BÁSICOS	2006	7	
9	3	4	1	8522104212	INTRODUÇÃO À ECONOMETRIA	1990	1	
10	3	5	2	0324113641	INTRODUCTORY ECONOMETRICS	2003	2	
11	3	2	1	8535213430	ECONOMETRIA - MODELOS E PREVISÕES	2004	1	
12	3	6	1	8502041231	ESTATÍSTICA E INTRODUÇÃO À ECONOM	2003	1	

Registro: 1 de 12 | Sem Filtro | Pesquisar

# Exercício: Consulta na base de dados



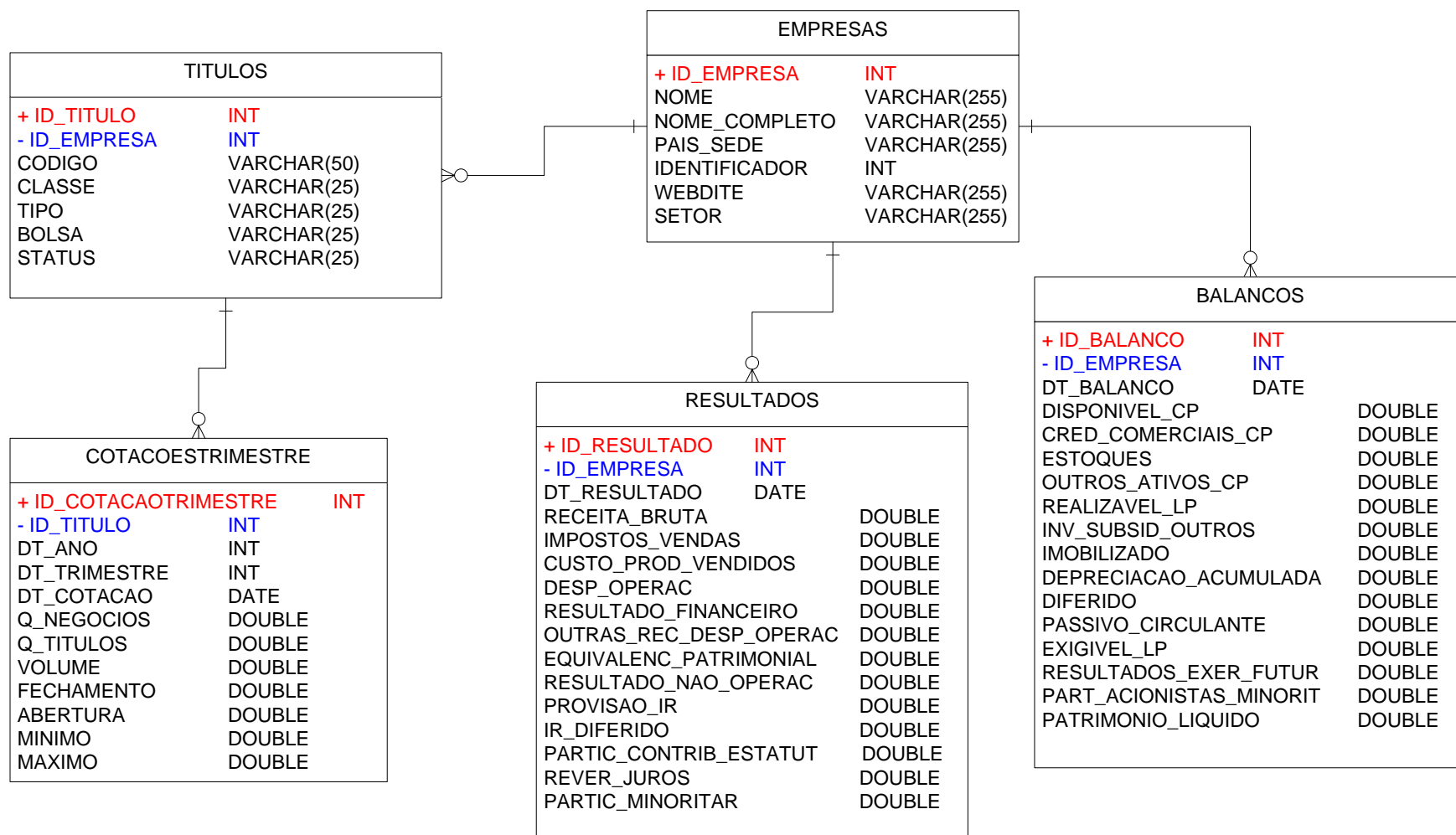
Informações básicas:

**Dados da Empresa:** Nome, País sede, Setor que atua

**Dados de Balanços:** Ativo, Passivo, Patrimônio Líquido

**Demonstrativos de Resultados:** Receita Bruta, Lucro Bruto, Lucro Operacional, Lucro Líquido

**Títulos Negociados e Cotações:** Código do título, classe (ON, PN...), cotação, data da cotação



# Tipos de Dados

## Tipos numéricos:

- **INT** ou **INTEGER**: Número inteiro  
Utilizada para criar Chaves primárias
- **FLOAT**: Número real (*floating-point*) “pequeno” com precisão simples
- **DOUBLE**: Número real com precisão dupla (*double-precision*).

Outros tipos numéricos comuns em bancos de dados: SMALLINT, NUMERIC, DECIMAL, REAL



# Tipos de Dados

## Tipos Texto

- **CHAR:** Tipo para armazenar cadeias de caracteres com tamanho fixo.
  - Deve-se informar o número máximo de caracteres suportado pelo campo.
  - Os caracteres faltantes serão preenchidos com 'espaços' à direita.
  - É comum o limite de 255 caracteres.
- **VARCHAR:** Armazena cadeias de caracteres (strings) de tamanho variável.
  - Necessariamente deve-se fornecer o tamanho máximo da sequência de caracteres que poderá ser armazenada.
  - Caso um registro tenha menos caracteres que o suportado pelo campo, os caracteres restantes ficarão vazios.
  - É comum o limite de 255 caracteres.
- **TEXT:** Tipo para armazenar cadeias de caracteres (textos) mais extensos.
  - Normalmente é possível especificar o tamanho máximo do texto que poderá ser armazenado mas não é obrigatório.

Outros tipos para armazenar texto: MEDIUMTEXT, LONGTEXT, BLOB.

# Tipos de Dados

## Tipos Data e Horário:

- **DATE**: Tipo campo para armazenar datas
- **DATETIME** e **TIMESTAMP**: Tipo para armazenar a combinação de data e horário. A diferença entre ambos, quando há, varia de acordo com a implementação de banco de dados.
- **TIME**: Tipo para armazenar horários
- **INTERVAL**: Tipo para armazenar intervalos de tempo.

# SQL - Structured Query Language

## SQL (*Structured Query Language*)

- Linguagem estruturada definida por regras de sintaxe e um vocabulário próprio
- Permite construir frases (*queries* ou comandos)
- Comandos interpretados por um Sistema Gerenciador de Banco de Dados que executará operações sobre o conjunto de dados.
- O conjunto de comando inclui operações para:
  - Criar uma estrutura de armazenamento e relacionamento entre dados
  - Modificar a estrutura criada
  - Operações básicas de manipulação de dados: selecionar (select), inserir (insert) , atualizar (update) , apagar (delete)
  - Indexação do conteúdo da base para tornar os operações mais rápidas
  - Gerenciamento do acesso dos usuários aos dados

## 1. Data Definition Language (DDL):

Criação de uma tabela: `CREATE TABLE`

Remoção de uma tabela: `DROP TABLE`

Altreração da estrutura da tabela: `ALTER TABLE`

Criação de um índice: `CREATE INDEX`

Remoção de um índice: `DROP INDEX`

Altreração de um índice: `ALTER INDEX`

## 2. Data Manipulation Language (DML):

Inserção de dados: `INSERT`

Atualização de dados: `UPDATE`

Remoção de dados: `DELETE`

# SQL - Structured Query Language

## 3. Data Query Language (DQL):

Seleção dos dados: `SELECT`

## 4. Data Control Language (DCL):

Criação de um usuário: `CREATE USER`

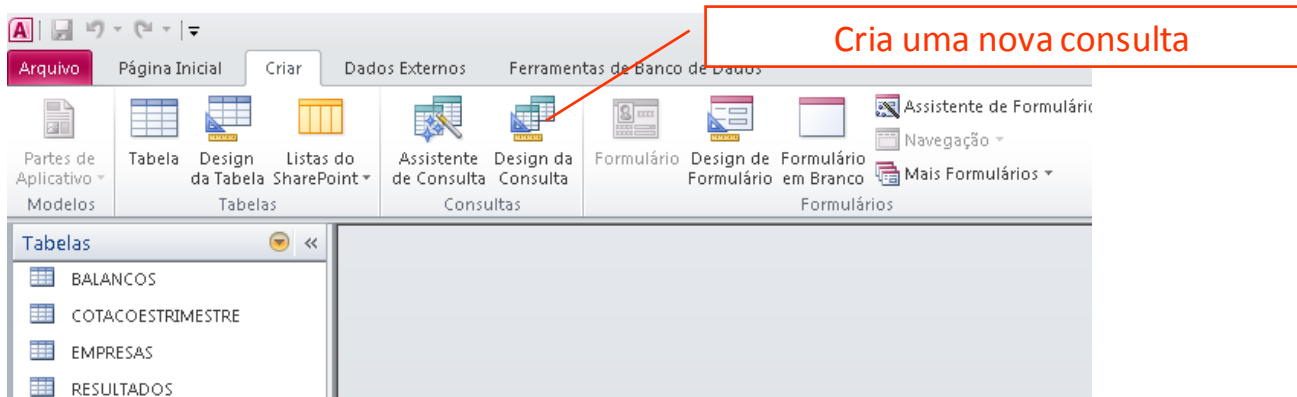
Alteração de um usuário: `ALTER USER`

Concessão de privilégios: `GRANT`

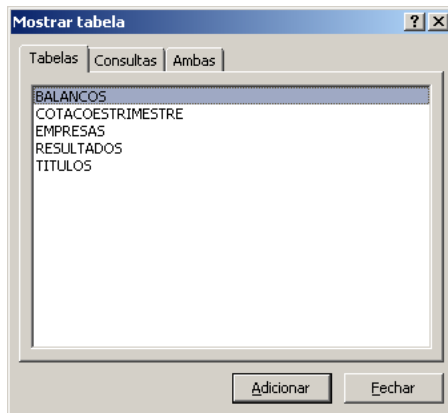
Revogação de privilégios: `REVOKE`

# Comandos SQL no MS Access

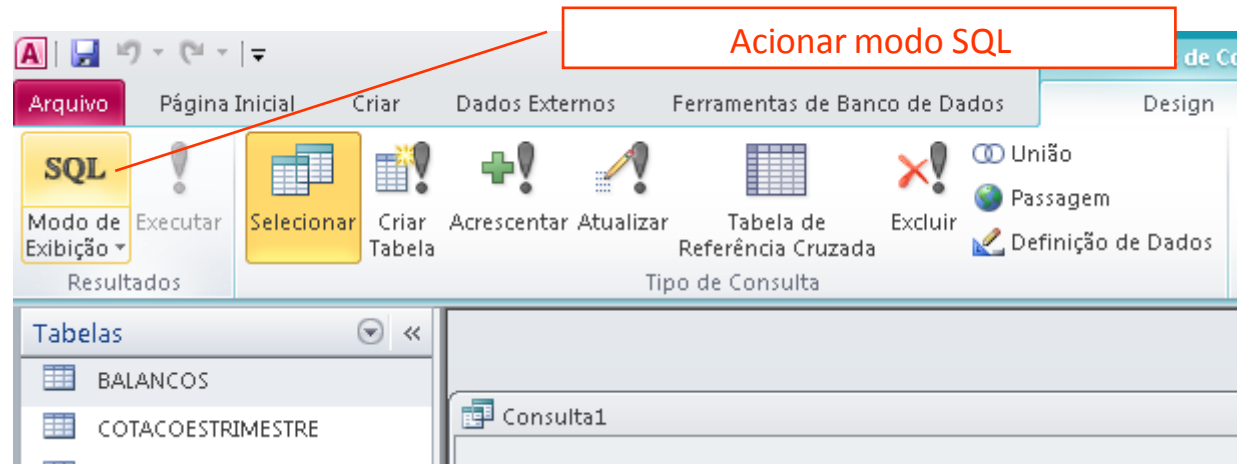
## 1. Abrir o assistem de consulta



## 2. Fechar a Janela “Mostrar Tabela”



## 3. Escolher o modo de exibição SQL



# SQL - *Data Definition Language* (DDL)

## Sintaxe do comando CREATE TABLE:

```
CREATE TABLE NomeTabela (
    NomeColuna1      TipoDoCampoColuna1 [ObrigaçõesColuna1],
    NomeColuna2      TipoDoCampoColuna2 [ObrigaçõesColuna2],
    NomeColuna3      TipoDoCampoColuna3 [ObrigaçõesColuna3],
    (...)
    NomeColunaN      TipoDoCampoColunaN [ObrigaçõesColunaN],

    [ObrigaçõesDaTabela]
)
```

## Sintaxe do comando DROP TABLE:

```
DROP TABLE NomeTabela
```

# Obrigações (*Constrains*)

## Objetivo:

- Garantir da integridade da base de dados e impor regras de negócio ao conjunto dos dados.
- Instruções adicionais fornecidas no momento da criação da tabela que impõe algumas regras obrigatórias para colunas da tabela ou para a tabela como um todo.
- Operações: inserção, atualização ou remoção de registros, se alguma obrigação for desrespeitada ocorre um erro e a operação é abortada.

## Principais *constrains* para colunas:

- PRIMARY KEY
- NOT NULL

Exemplo: Cadastro de clientes evita que seja feito um registro com nome do cliente vazio.

- UNIQUE

Exemplo: Coluna de CPF do cadastro de pessoas para evitar 2 pessoas com mesmo CPF.

## Principais *constrains* para tabela:

- FOREIGN KEY:
  - Vincula campos de chave estrangeira de uma tabela com a chave primária de outra tabela.
  - Garantir a integridade relacional



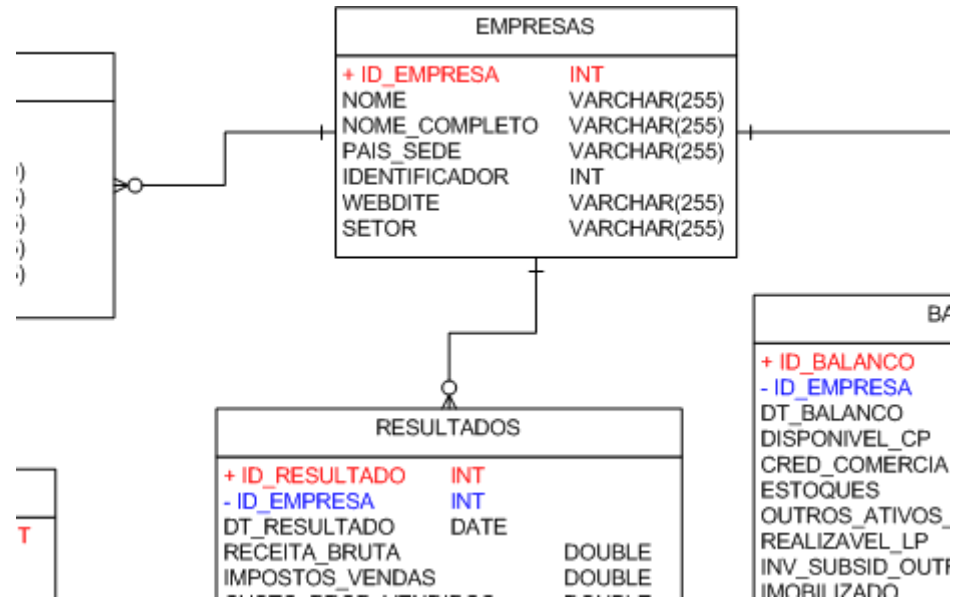
# Exemplos: Create Table

Tabela para cadastro de Empresas:

ID\_EMPRESA: Chave Primária

NOME: Único

```
CREATE TABLE EMPRESAS (
  ID_EMPRESA      INT PRIMARY KEY,
  NOME            VARCHAR(255) UNIQUE,
  NOME_COMPLETO   VARCHAR(255) UNIQUE,
  PAIS_SEDE       VARCHAR(255),
  IDENTIFICADOR   INT,
  WEBSITE         VARCHAR(255),
  SETOR           VARCHAR(255)
);
```

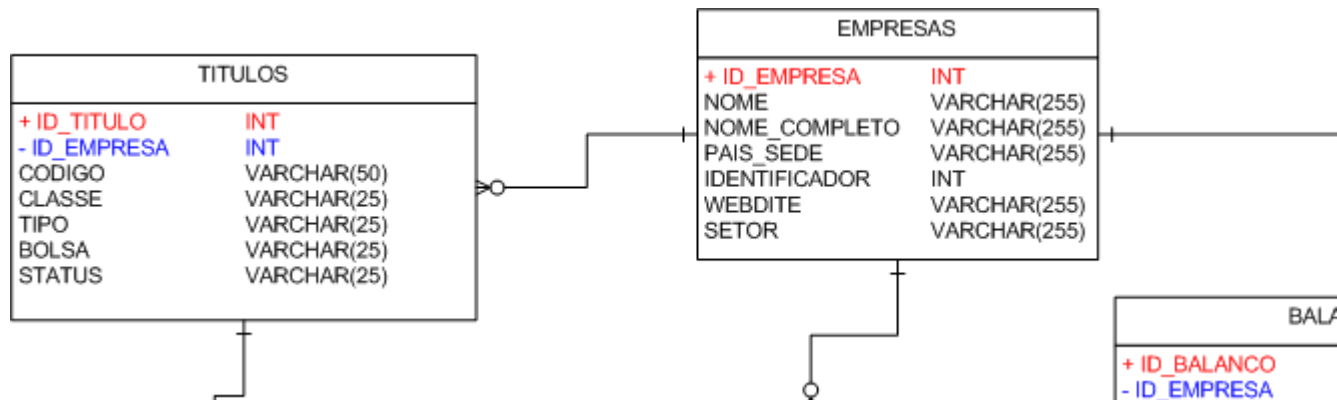


# Exemplos: Create Table

Tabela para cadastro de Livros:

ID\_EMPRESA: Chaves estrangeira

```
CREATE TABLE TITULOS (
  ID_TITULO      INT PRIMARY KEY,
  ID_EMPRESA     INT,
  CODIGO         VARCHAR(50),
  CLASSE        VARCHAR(25),
  TIPO           VARCHAR(25),
  BOLSA         VARCHAR(25),
  STATUS        VARCHAR(25),
  CONSTRAINT FK_TITULO_EMPRESA FOREIGN KEY (ID_EMPRESA) REFERENCES EMPRESAS
);
```



# SQL - *Data Manipulation Language* (DML)

## Sintaxe do comando **INSERT**:

```
INSERT INTO NomeTabela (NomeColuna_A, NomeColuna_B, ...)
VALUES (ValorColuna_A, ValorColuna_B, ...)
```

- No comando **INSERT** não há a obrigatoriedade de fornecer todos os campos da tabela, a menos dos campos que tenham restrições ou regras específicas, exemplo: **NOT NULL**.
- Cadeias de caracteres (*strings*) nas expressões SQL são delimitadas por aspas simples `''`.

## Exemplos:

- Inserir uma nova empresa na tabela **EMPRESAS**:

```
INSERT INTO EMPRESAS (ID_EMPRESA, NOME, NOME_COMPLETO,
PAIS_SEDE, IDENTIFICADOR, WEBSITE, SETOR) VALUES (1000, 'Uma
Nova Empresa', 'Uma Nova Empresa S/A', 'Brasil', 2000,
'www.novaempresa.com.br', 'Metalurgia');
```

- Tentando violar a restrição: **NOME** tem que ser único

```
INSERT INTO EMPRESAS (ID_EMPRESA, NOME, NOME_COMPLETO,
PAIS_SEDE, IDENTIFICADOR, WEBSITE, SETOR) VALUES (1001, 'Uma
Nova Empresa', 'Uma Nova Empresa LTDA', 'EUA', 2001,
'www.novaempresa.com.br', 'Mineração');
```

# SQL - *Data Manipulation Language* (DML)

## Sintaxe do comando UPDATE:

```
UPDATE NomeTabela
    SET NomeColuna_A=ValorColuna_A, NomeColuna_B=ValorColuna_B...
    [WHERE CondiçãoDeSeleção AND | OR
        CondiçãoDeSeleção...]
```

## Sintaxe do comando DELETE:

```
DELETE FROM NomeTabela
    [WHERE CondiçãoDeSeleção AND | OR
        CondiçãoDeSeleção...]
```

- A cláusula `WHERE` nos comandos `UPDATE` e `DELETE` é opcional e utilizada para restringir os registros que serão afetados pelo comando
- Quando a cláusula `WHERE` é omitida os comandos afetarão **TODOS** os registros
- Operadores de igualdade (`'='`), desigualdade (`'<'`, `'>'`, `'<>'`), expressões e funções especiais (`LIKE` e `IS NULL`).
- Cadeias de caracteres (*strings*) nas expressões SQL são delimitadas por aspas simples `'`.

# SQL - Data Manipulation Language (DML)

## Exemplos:

- Atualizar EMPRESAS informando um novo nome para o registro ID\_EMPRESA= 1000:

```
UPDATE EMPRESAS SET NOME = 'Novo Nome da Empresa' WHERE
ID_EMPRESA=1000;
```

- Apagar da tabela EMPRESAS o registro no qual o NOME é 'Novo Nome da Empresa':

```
DELETE FROM EMPRESAS WHERE NOME = 'Novo Nome da Empresa';
```

# SQL - *Data Query Language* (DQL)

## Sintaxe do comando SELECT:

```
SELECT [* | NomeTabela.Campo, NomeTabela.Campo, ...]
      FROM NomeTabela, NomeTabela, ....
      [WHERE
          CondiçãoDeSeleção AND | OR
          CondiçãoDeSeleção...]
      [ORDER BY NomeTabela.Campo, NomeTabela.Campo... [DESC]]
      [GROUP BY NomeTabela.Campo]
```

- A cláusula **SELECT** especifica os campos que devem formar o resultado da consulta
  - O caractere '\*', indica que todos os campos devem ser retornados
  - Colocar o nome da tabela antes do nome do campo é obrigatório quando estamos consultando ao mesmo tempo tabelas que tenham campos com o mesmo nome
- Na cláusula **FROM** devem constar todas as tabelas cujos campos foram mencionados em qualquer cláusula do comando de consulta.
- A cláusula **WHERE** é opcional, quando não definida retorna como resultado da consulta todos dos registros da tabela.

# WHERE: Critérios de Seleção

## Operadores Básico

- Utilizados para campos numéricos, datas e campos texto
- Data: ordem cronológica, sendo que quando mais recente, maior.

Por exemplo, '2006-02-01 18:50:00' é maior que "2005-03-01 00:00:00".

- Texto: ordem lexicográfica.

Operador	Descrição	Exemplo
=	igual	ID_EMPRESA=2; NOME='Petrobras'
<> , !=	diferente	SETOR<>'Textil'; PAIS_SEDE<>'Brasil'
>	maior	BALANCOS.DT_BALANCO>31/12/2002
>=	maior ou igual	BALANCOS.DISPONIVEL_CP>1000000;
<	menor	BALANCOS.DISPONIVEL_CP<1000000;
<=	menor ou igual	BALANCOS.ESTOQUES>1000000;

## Operadores Lógicos

Operador	Significado	Exemplo
AND	E	DT_BALANCO>31/12/2002 AND ID_EMPRESA=1
OR	OU	DISPONIVEL_CP>10000 OR ESTOQUES > 2000000

# WHERE: Critérios de Seleção

## Operadores Especiais

Operador	Descrição
<b>LIKE</b>	Busca por palavras ou parte de palavras
<b>BETWEEN</b>	Determina um intervalo de busca pode ser usado para números e datas
<b>IN</b>	Determina uma lista de valores
<b>IS [NOT] NULL</b>	Retringe a procura a campos [não] nulos

## Sintaxe do operador **LIKE**:

```
SELECT [* | NomeTabela.NomeCampo,...]
FROM NomeTabela,....
WHERE NomeCampo LIKE 'Expressão'
```

Exemplo	Interpretação
LIKE 'ECONOMIA'	Registros com exatamente a palavra 'ECONOMIA'
LIKE 'ECONO%'	Registros que começam com a palavra 'ECONO' exemplo: 'ECONOMETRIA'
LIKE '%ECONOMIA'	Registros que terminam com a palavra 'ECONOMIA' exemplo: 'MACROECONOMIA, 'MICROECONOMIA'
LIKE '%ECONO%'	Registros com a palavra 'ECONO' em qualquer posição exemplo: 'MACROECONOMIA, 'MICROECONOMIA' ECONOMETRIA



# WHERE: Critérios de Seleção

## Sintaxe do operador IN:

```
SELECT [* | NomeTabela.NomeCampo, ...]
FROM NomeTabela, ....
WHERE NomeCampo IN (ListaDeValores)
```

### Exemplo:

Para selecionar registros cujo ANO seja 2001 ou 2004:

```
(...) WHERE (...) ANO IN (2001,2004)
```

## Sintaxe do operador IS [NOT] NULL:

```
SELECT [* | NomeTabela.NomeCampo, ...]
FROM NomeTabela, ....
WHERE NomeCampo IS [NOT] NULL
```

### Exemplo:

Para selecionar registros cuja DATA\_CADASTRO não seja vazio:

```
(...) WHERE (...) DATA_CADASTRO IS NOT NULL
```

# WHERE: Critérios de Seleção

## Sintaxe do operador BETWEEN:

```
SELECT [* | NomeTabela.NomeCampo,...]
FROM NomeTabela,....
WHERE NomeCampo BETWEEN LimiteInferior AND LimiteSuperior
```

## Exemplo:

Para selecionar registros cujas datas de cadastro estejam no ano de 2001:

```
(...) WHERE (...)
        DATA_CADASTRO BETWEEN '2001-01-01' AND '2001-12-13'
```

# WHERE: Critérios de Seleção

Exemplos Básico:

**Selecionar da tabela EMPRESAS todos dos campos da empresa com ID\_EMPRESA=1**

```
SELECT * FROM EMPRESAS WHERE ID_EMPRESA = 1
```

**Selecionar da tabela EMPRESAS os campos ID\_EMPRESA, NOME, PAIS\_SEDE das empresas do setor 'Textil'**

```
SELECT ID_EMPRESA, NOME, PAIS_SEDE FROM EMPRESAS WHERE SETOR =  
'Textil'
```

**Selecionar da tabela EMPRESAS os campos ID\_EMPRESA, NOME, PAIS\_SEDE, e o SETOR das empresas do setor 'Textil' OU do setor de 'Mineração'**

```
SELECT ID_EMPRESA, NOME, PAIS_SEDE FROM EMPRESAS WHERE SETOR =  
'Textil' or SETOR = 'Mineração'
```

# WHERE: Critérios de Seleção

## Exemplos Básico:

**Selecionar da tabela BALANCOS os campos ID\_EMPRESA, DT\_BALANCO e DISPONIVEL\_CP para os registros da ID\_EMPRESA = 1 e datas superiores a 01/01/2000**

```
SELECT ID_EMPRESA, DISPONIVEL_CP, DT_BALANCO FROM BALANCOS WHERE
ID_EMPRESA = 1 AND DT_BALANCO > 01/01/2000
```

**Selecionar da tabela BALANCOS os campos ID\_EMPRESA, DT\_BALANCO e DISPONIVEL\_CP para os registros da ID\_EMPRESA = 1 e datas superiores a 01/01/2000 e inferiores a 31/12/2006**

```
SELECT ID_EMPRESA, DISPONIVEL_CP, DT_BALANCO FROM BALANCOS WHERE
ID_EMPRESA = 1 AND DT_BALANCO > 01/01/2000 AND DT_BALANCO <
31/12/2006
```

# Acesso a Banco de Dados e Queries em R

Para acessar Bancos de Dados em R e executar queries podemos:

1. Utilizar pacotes e drives específicos por exemplo para MySQL, Oracle, SQL Server, etc...

Acesso é feito diretamente ao banco de dados passando os parâmetros da conexão

2. Utilizar os pacotes RODBCC e RJDBC/Bridge ODBC

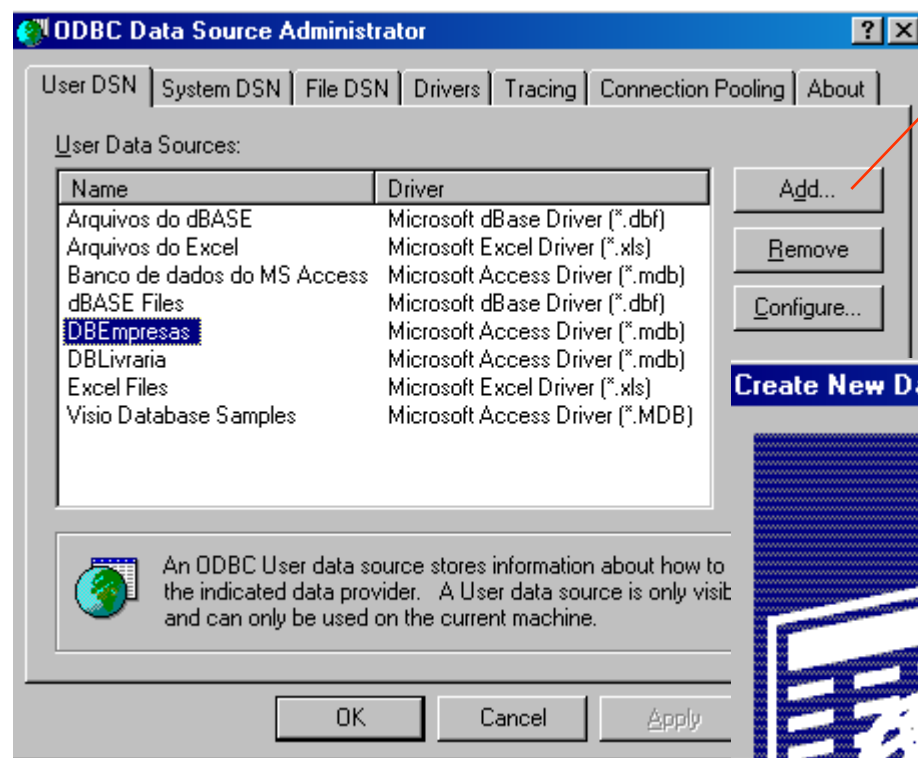
É necessário criar no sistema operacional uma DSN (“Data Source Names”)

3. Para o MS Access e MS Excel não é necessário criar DSNs pois o pacote RODBCC pode acessar diretamente estas bases.

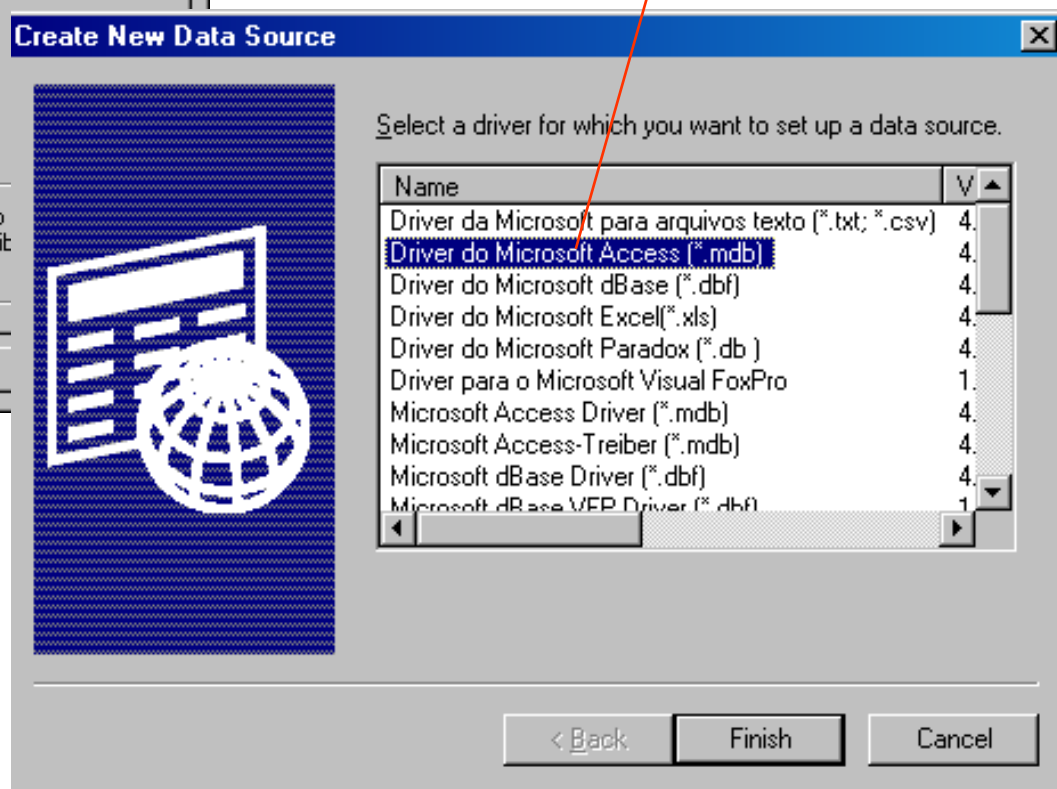
# Conexão com Banco de Dados - DNS

Para criar uma nova conexão ODBC;

Clicar no botão "Adicionar"



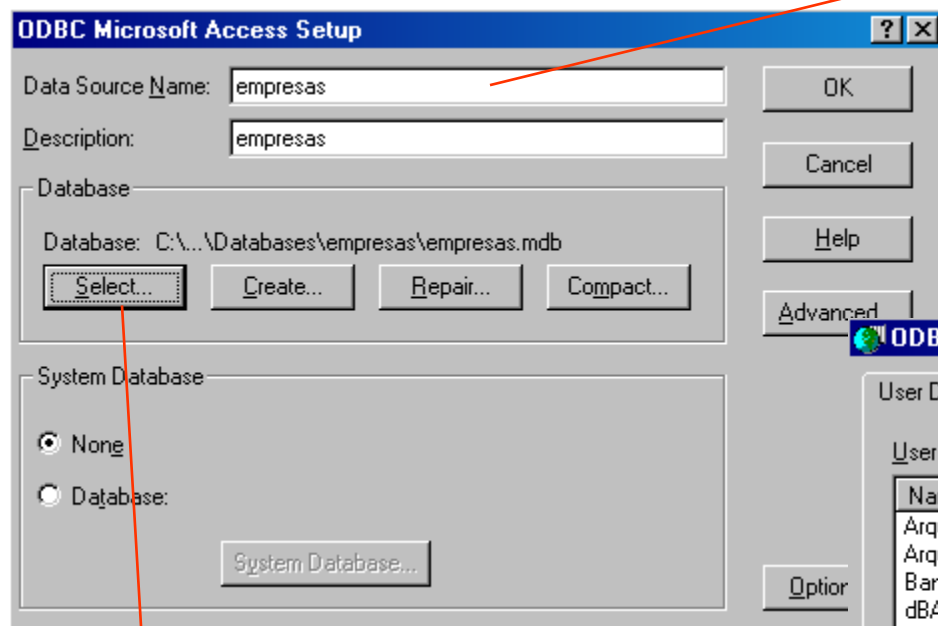
Selecionar o tipo de banco de dados



# Conexão com Banco de Dados - DSN

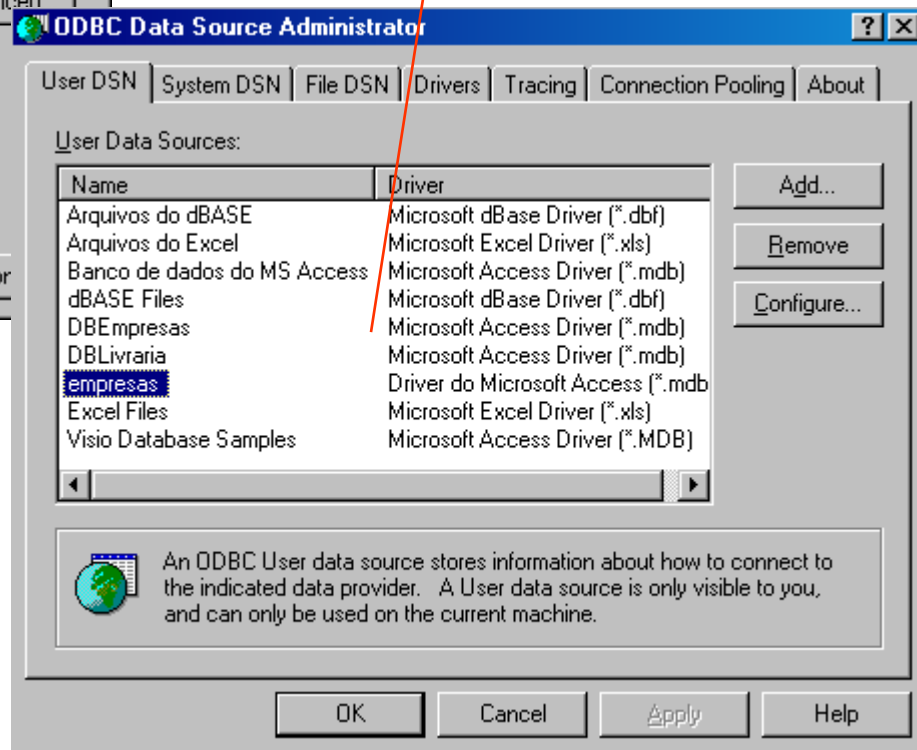
Para criar uma nova conexão ODBC:

1. Escolher um nome



2. Selecionar o arquivo

Conexão criada



# Conexão ao *MS Access* - Queries em R

## Pacote RODBAC:

- Conexão DSN Genérico

```
odbcConnect(dsn, uid = "", pwd = "", ...)
```

- Conexão ao Access

```
odbcConnectAccess(access.file, uid = "", pwd = "", ...)
```

```
odbcConnectAccess2007(access.file, uid = "", pwd = "", ...)
```



# Conexão ao *MS Access* - Queries em R

Abre a conexão com a base de dados e executa uma query *select*:

```
library(RODBC);
DataLoaderSQL= function(db, query){

  ## Estabelece a conexão com o banco de dados
  conn = odbcConnectAccess(db$filename);

  ## Executa a query; rs = ResultSet
  rs = sqlQuery(conn,query);

  ## Fecha a conexão
  odbcClose(conn);

  ## Retorna o dado par a programa principal
  return(rs);
}
```

# Conexão ao *MS Access* - Queries em R

Abre a conexão com a base de dados e executa uma query *select*:

```
## Teste da função de carga de dados
```

```
## Carrega a biblioteca
```

```
source("../lib/DataLoader.r");
```

```
## Parâmetros de acesso ao banco de dados
```

```
db = list(filename="../database/empresas.mdb", user="", passwd="");
```

```
## Define a query
```

```
query = "select * from EMPRESAS";
```

```
## Executa a query
```

```
data = DataLoaderSQL(db,query);
```

```
## Mostra os dados
```

```
print(data);
```

# WHERE: Critérios de Seleção

Exemplos Básico:

**Selecionar da tabela EMPRESAS todos os campos das empresas que tem a palavra “Banco” no campo NOME\_COMPLETO**

```
SELECT * FROM EMPRESAS WHERE NOME_COMPLETO Like "*BANCO*"
```

**Selecionar da tabela EMPRESAS todos os campos das empresa que estejam cadastradas com “S.A.”, ou seja, que tenham “S.A.” no campo NOME\_COMPLETO**

```
SELECT * FROM EMPRESAS WHERE NOME_COMPLETO Like "*S.A.*"
```

**Selecionar da tabela BALANCOS os campos ID\_EMPRESA, DT\_BALANCO, ESTOQUES e CRED\_COMERCIAIS para os balanços publicados entre 1996 e 2000, inclusive. Utilizar a função YEAR e os operadores de desigualdade.**

```
SELECT ID_EMPRESA, DT_BALANCO, CRED_COMERCIAIS_CP, ESTOQUES
FROM BALANCOS WHERE YEAR(DT_BALANCO) >= 1996 AND
YEAR(DT_BALANCO) <= 2000
```

# WHERE: Critérios de Seleção

## Exemplos Básico:

**Selecionar da tabela BALANCOS os campos ID\_EMPRESA, DT\_BALANCO, ESTOQUES e CRED\_COMERCIAIS para os balanços publicados entre 1996 e 2000, inclusive. Utilizar a função YEAR o comando BETWEEN / AND**

```
SELECT ID_EMPRESA, DT_BALANCO, CRED_COMERCIAIS_CP, ESTOQUES FROM
BALANCOS WHERE YEAR(DT_BALANCO) BETWEEN 1996 AND 2000
```

**Selecionar da tabela BALANCOS os campos ID\_EMPRESA, DT\_BALANCO, ESTOQUES e CRED\_COMERCIAIS para os balanços publicados entre 1996 e 2000, inclusive. Utilizar a função YEAR o comando IN**

```
SELECT ID_EMPRESA, DT_BALANCO, CRED_COMERCIAIS_CP, ESTOQUES FROM
BALANCOS WHERE YEAR(DT_BALANCO) IN (1996,1997,1998,1999, 2000)
```

**Selecionar da tabela BALANCOS os campos ID\_EMPRESA, DT\_BALANCO, ESTOQUES e CRED\_COMERCIAIS para os balanços publicados entre 1996 e 2000, inclusive, para os quais os campos ESTOQUES e CRED\_COMERCIAIS não sejam vazios**

```
SELECT ID_EMPRESA, DT_BALANCO, CRED_COMERCIAIS_CP, ESTOQUES FROM
BALANCOS WHERE YEAR(DT_BALANCO) IN (1996,1997,1998,1999, 2000) AND
CRED_COMERCIAIS_CP IS NOT NULL AND ESTOQUES IS NOT NULL
```

## Expressões Aritméticas:

É possível colocar expressões aritméticas tanto nos campos selecionados quanto nos critérios.

Exemplo:

Selecione da tabela BALANCOS o ID\_EMPRESA data e a relação passivo circulante e exigível de longo prazo quando esta razão 1.

```
SELECT
    ID_EMPRESA, DT_BALANCO, PASSIVO_CIRCULANTE/EXIGIVEL_LP
FROM
    BALANCOS
WHERE
    ID_EMPRESA = 1 AND PASSIVO_CIRCULANTE/EXIGIVEL_LP > 1
```

# Group By e Order By

**GROUP BY** :opcional e deve ser utilizada para agrupar registros que tenham alguma característica em comum e normalmente é utilizada com outras funções especiais como por exemplo:

**SUM**: Função que soma um determinado campo dos todos os registros que foram agrupados

**CONT**: Função que conta os registros que foram agrupados

**ORDER BY**: opcional e é utilizada para ordenar o resultado da consulta por um ou mais campos que necessariamente devem constar no resultado.

# WHERE: Critérios de Seleção

## Exemplos Básico:

**Selecionar da tabela BALANCOS os campos ID\_EMPRESA, DT\_BALANCO, ESTOQUES e CRED\_COMERCIAIS para os balanços publicados entre 1996 e 2000, inclusive ordenando os resultados pelos campos ID\_EMPRESA e DT\_BALANCO.**

```
SELECT ID_EMPRESA, DT_BALANCO, CRED_COMERCIAIS_CP, ESTOQUES FROM
BALANCOS WHERE YEAR(DT_BALANCO) >= 1996 AND YEAR(DT_BALANCO) <= 2000
order by ID_EMPRESA, DT_BALANCO
```

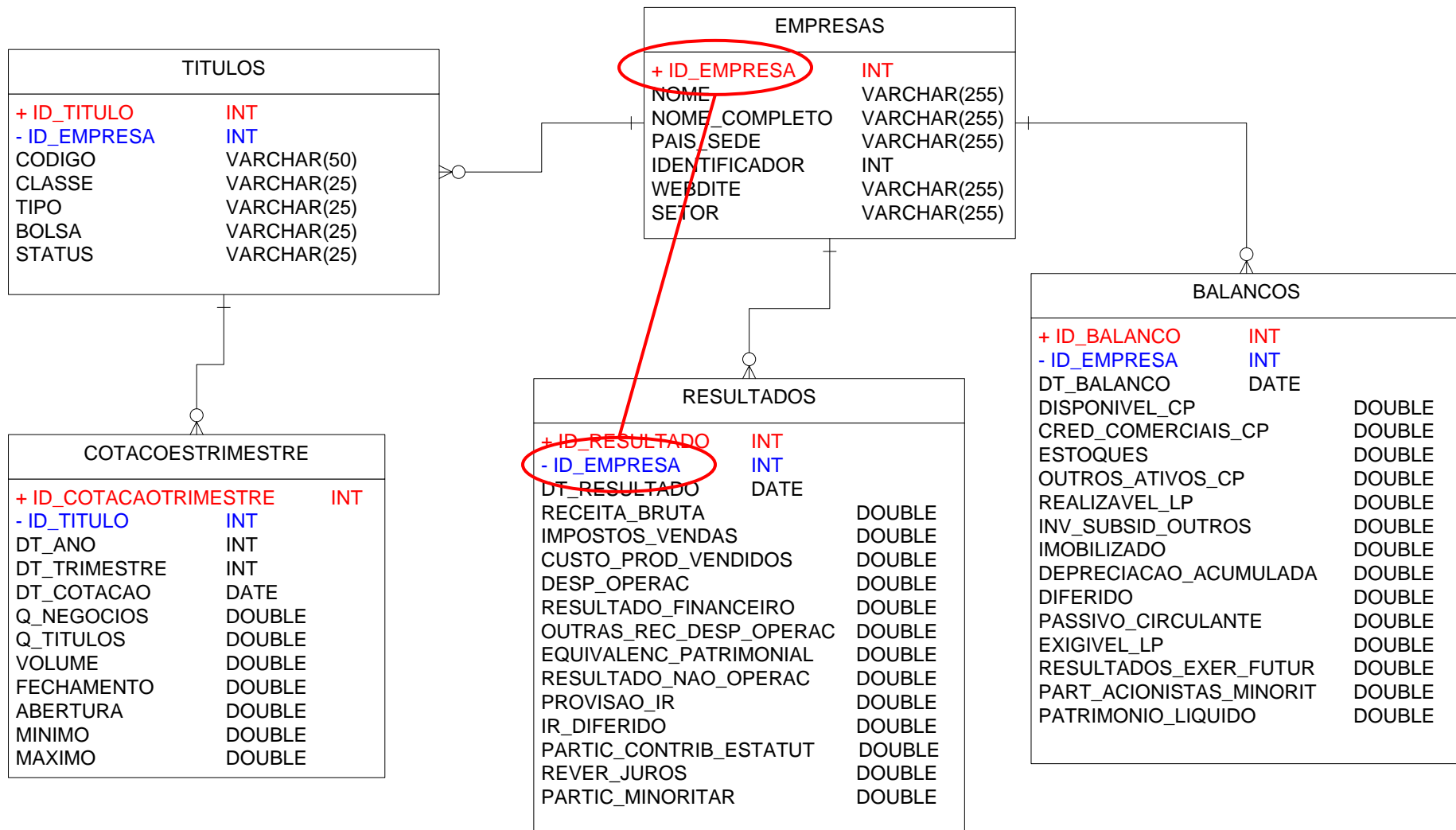
**Calcular a receita bruta acumulada entre os anos de 2004 e 2005 para a empresas, ordenando o resultado e ordem decrescente. Utilizar os operadores SUM e GROUP BY.**

```
SELECT ID_EMPRESA, sum(RECEITA_BRUTA) FROM RESULTADOS WHERE
YEAR(DT_RESULTADO) >= 2004 AND YEAR(DT_RESULTADO) <= 2005 GROUP BY
ID_EMPRESA ORDER BY sum(RECEITA_BRUTA) DESC
```

```
SELECT ID_EMPRESA, sum(RECEITA_BRUTA) AS RB_ACUMULADA FROM RESULTADOS
WHERE YEAR(DT_RESULTADO) >= 2004 AND YEAR(DT_RESULTADO) <= 2005 GROUP
BY ID_EMPRESA ORDER BY sum(RECEITA_BRUTA) DESC
```

# Consultas envolvendo mais de uma tabela

**Exemplo:** Selecionar a série histórica de receitas brutas da Petrobrás.



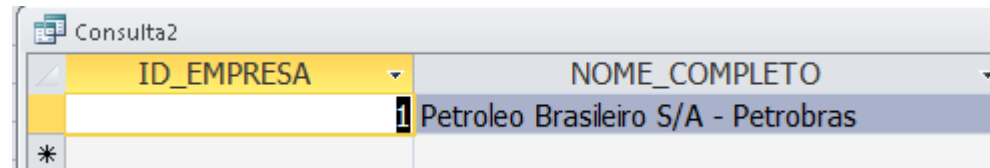


# Consultas envolvendo mais de uma tabela

**Exemplo:** Selecionar a série histórica de receitas brutas da Petrobrás.

**1o Passo:** Descobrir o ID\_EMPRESA para a Petrobrás

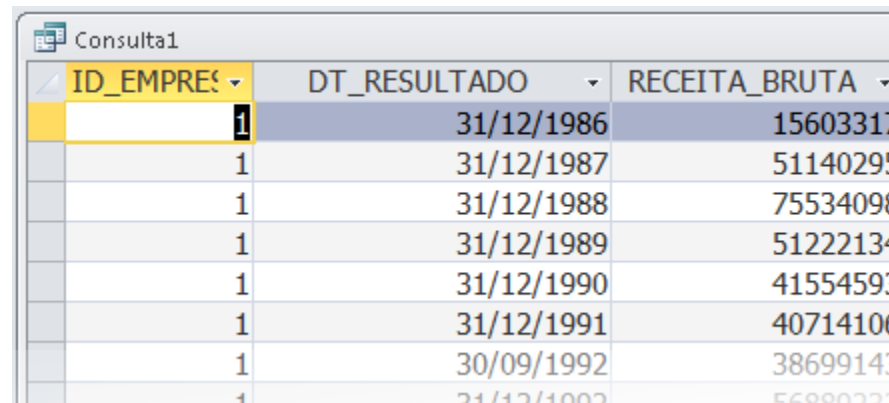
```
SELECT ID_EMPRESA, NOME_COMPLETO FROM EMPRESAS WHERE
NOME_COMPLETO LIKE "*PETROBR*"
```



ID_EMPRESA	NOME_COMPLETO
1	Petroleo Brasileiro S/A - Petrobras

**2o Passo:** Selecionar da tabela RESULTADOS a série de receitas brutas para os registros no  
quais ID\_EMPRESA = 1

```
SELECT ID_EMPRESA, DT_RESULTADO, RECEITA_BRUTA FROM RESULTADOS
WHERE ID_EMPRESA=1 ORDER BY DT_RESULTADO
```



ID_EMPRESA	DT_RESULTADO	RECEITA_BRUTA
1	31/12/1986	15603317
1	31/12/1987	51140295
1	31/12/1988	75534098
1	31/12/1989	51222134
1	31/12/1990	41554593
1	31/12/1991	40714106
1	30/09/1992	38699143
1	31/12/1992	56880777

# Consultas envolvendo mais de uma tabela

**Exemplo:** Selecionar a série histórica de receitas brutas da Petrobrás.

**Único Passo:** Selecionar da tabela RESULTADOS o campo RECEITA\_BRUTA dos registros cujo ID\_EMPRESA seja igual ao ID\_EMPRESA dos registros da tabela EMPRESAS cujo campo NOME seja igual a 'Petrobras'

SELECT

**EMPRESAS.ID\_EMPRESA, NOME, DT\_RESULTADO, RECEITA\_BRUTA**

FROM

**EMPRESAS, RESULTADOS**

WHERE

**RESULTADOS.ID\_EMPRESA=EMPRESAS.ID\_EMPRESA**

AND

**EMPRESAS.NOME='Petrobras'**

**ORDER BY DT\_RESULTADO**

Campos em tabelas diferentes com o mesmo nome devem identificados como:  
***NomeDaTabela.NomeDoCampo***

Incluir todas as tabelas envolvidas na consulta

Relacionar as chaves das tabelas

# Consultas envolvendo mais de uma tabela

**Exemplo:** Selecionar a série histórica de receitas brutas da Petrobrás.

```
SELECT EMPRESAS.ID_EMPRESA, NOME, DT_RESULTADO, RECEITA_BRUTA FROM
EMPRESAS, RESULTADOS WHERE RESULTADOS.ID_EMPRESA=EMPRESAS.ID_EMPRESA AND
EMPRESAS.NOME='Petrobras' ORDER BY DT_RESULTADO
```

ID_EMPRESA	DT_RESULTADO	RECEITA_BRUTA
1	31/12/1986	15603317
1	31/12/1987	51140295
1	31/12/1988	75534098
1	31/12/1989	51222134
1	31/12/1990	41554593
1	31/12/1991	40714106
1	30/09/1992	38699143
1	31/12/1992	56889233
1	31/12/1993	62285736
1	31/12/1994	55018499
1	31/12/1995	48914100
1	31/12/1996	51187775
1	31/12/1997	54490016
1	31/12/1998	52258858
1	31/03/1999	12965373
1	30/06/1999	28380136
1	30/09/1999	47776802
1	31/12/1999	66772201

# Consultas envolvendo mais de uma tabela

Exemplos Básico:

**Selecionar da tabela BALANCOS os campos DT\_BALANCO, DISPONIVEL\_CP, CRED\_COMERCIAIS\_CP, ESTOQUES, OUTROS\_ATIVOS\_CP para a Petrobras para os balanços publicados em 2005**

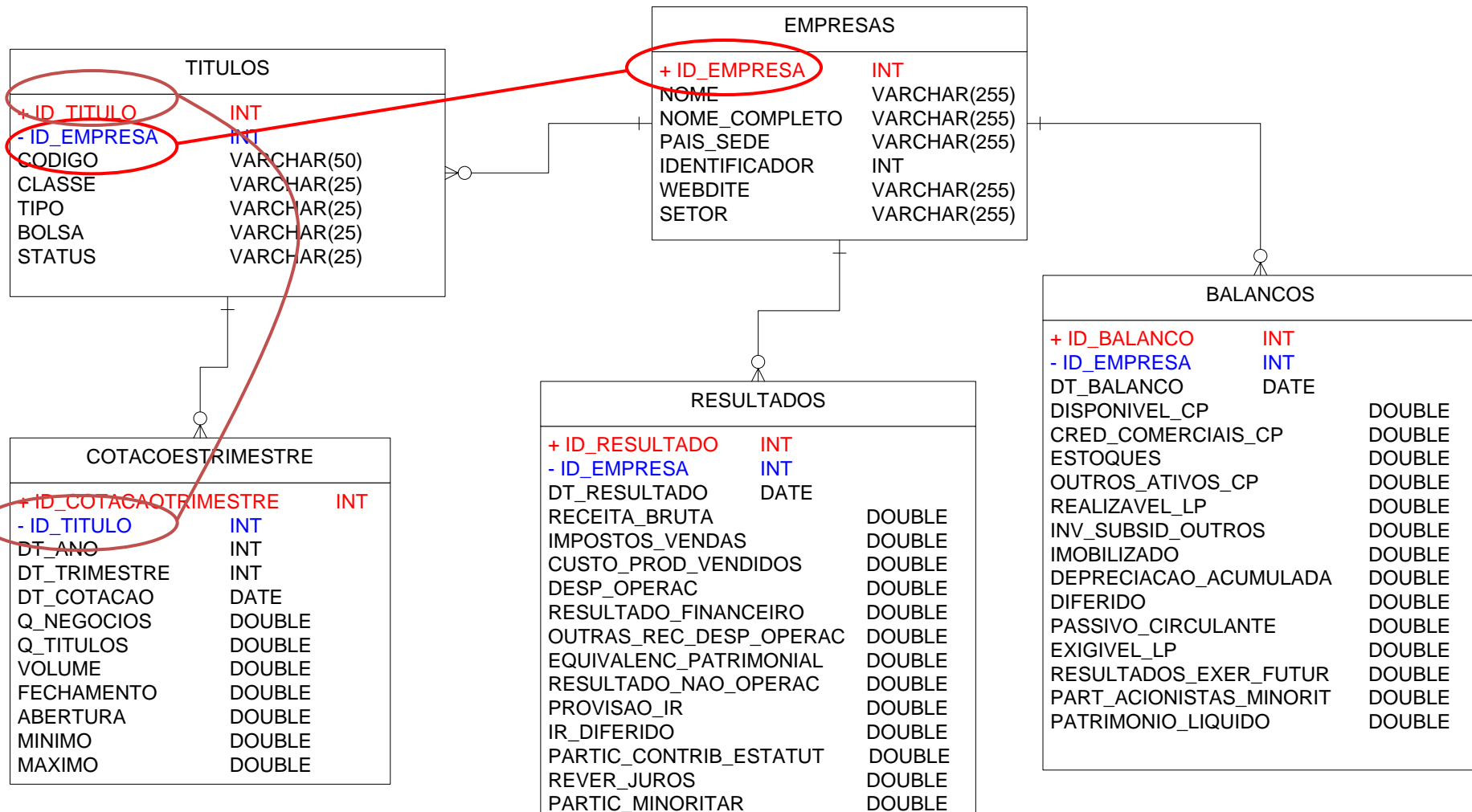
```
SELECT DT_BALANCO, DISPONIVEL_CP, CRED_COMERCIAIS_CP, ESTOQUES,
OUTROS_ATIVOS_CP FROM EMPRESAS, BALANCOS WHERE EMPRESAS.ID_EMPRESA =
BALANCOS.ID_EMPRESA AND NOME='Petrobras' and YEAR(DT_BALANCO)=2005
```

**Selecionar da tabela EMPRESAS o ID\_EMPRESA e o NOME, e da tabela BALANCOS os campos DT\_BALANCO, DISPONIVEL\_CP, CRED\_COMERCIAIS\_CP, ESTOQUES, OUTROS\_ATIVOS\_CP para os balanços publicados em 2005 para cada empresa cadastrada ordenando pelo campo ID\_EMPRESA.**

```
SELECT EMPRESAS.ID_EMPRESA, NOME, DT_BALANCO, DISPONIVEL_CP,
CRED_COMERCIAIS_CP, ESTOQUES, OUTROS_ATIVOS_CP FROM EMPRESAS, BALANCOS
WHERE EMPRESAS.ID_EMPRESA = BALANCOS.ID_EMPRESA AND
YEAR(DT_BALANCO)=2005 ORDER BY EMPRESAS.ID_EMPRESA
```

# Consultas envolvendo mais de uma tabela

**Exemplo:** Selecionar a cotações de todos os títulos da Petrobras retornando no código do título, a data da cotação e o valor de fechamento para o ano de 2005.



# Consultas envolvendo mais de uma tabela

**Exemplo:** Selecionar a cotações de todos os títulos da Petrobras retornando no código do título, a data da cotação e o valor de fechamento para o ano de 2005.

```

SELECT
    NOME, CODIGO, DT_COTACAO, FECHAMENTO
FROM
    EMPRESAS, TITULOS, COTACOESTRIMESTRE
WHERE
    EMPRESAS.ID_EMPRESA=TITULOS.ID_EMPRESA
    AND
    TITULOS.ID_TITULO=COTACOESTRIMESTRE.ID_TITULO
    AND
    NOME='Petrobras'
    AND
    DT_ANO = 2005

```

# Exercício

## Exercício 01

**Criar a função:**

**LiquidezCorrentePorPeriodo(company, dateBegin, dateEnd)**

**Onde**

**company:** nome da empresa

**dateBegin:** Data de Início do período

**dateEnd:** Data de final de período

**Retorna**

**Data Frame com as colunas:**

**period:** Data ao do final do período ao qual se refere o balanço

**liquidezCorrente:** Liquidez corrente do período