



Fundamentos de Business Intelligence e Análise de Dados

BLOCO: B.I. E ANÁLISE DE DADOS

PROF. RODRIGO EIRAS, M.SC.

[ETAPA 4] AULAS 1 E 2 – BANCOS DE DADOS, SGBDS E LINGUAGEM SQL



Na aula anterior...

- Correção do exercício extra
- Mais sobre relacionamentos na modelagem relacional
- Normalização



Agenda

- Fundamentos de Bancos de Dados
- O que são SGBDs
- Modelos suportados em SGBDs
- Funções de um SGBD
- Linguagem SQL
- Exemplos Práticos usando MySQL
- Exercícios

Bancos de dados

Os Bancos de Dados fazem parte do nosso dia-a-dia:

- Operação bancária
- Reserva de hotel
- Matrícula em uma disciplina da universidade
- Cadastro na vídeo locadora



Bancos de dados

Dado: fato do mundo real que está registrado

- Exemplos: endereço, data

Informação: fato útil que pode ser extraído direta ou indiretamente a partir dos dados

- Exemplos: endereço de entrega, idade

Banco de Dados (BD): coleção de dados inter-relacionados e persistentes que representa um sub-conjunto dos fatos presentes em um domínio de aplicação (universo de discurso)



Considere o contexto de uma grande organização que NÃO utiliza BD

- exemplo: domínio da Universidade
 - várias divisões gerenciais (com suas aplicações)
 - grande volume de dados
 - aplicações manipulam dados comuns

Acadêmica

Alunos
Professores
Disciplinas
Turmas
Salas

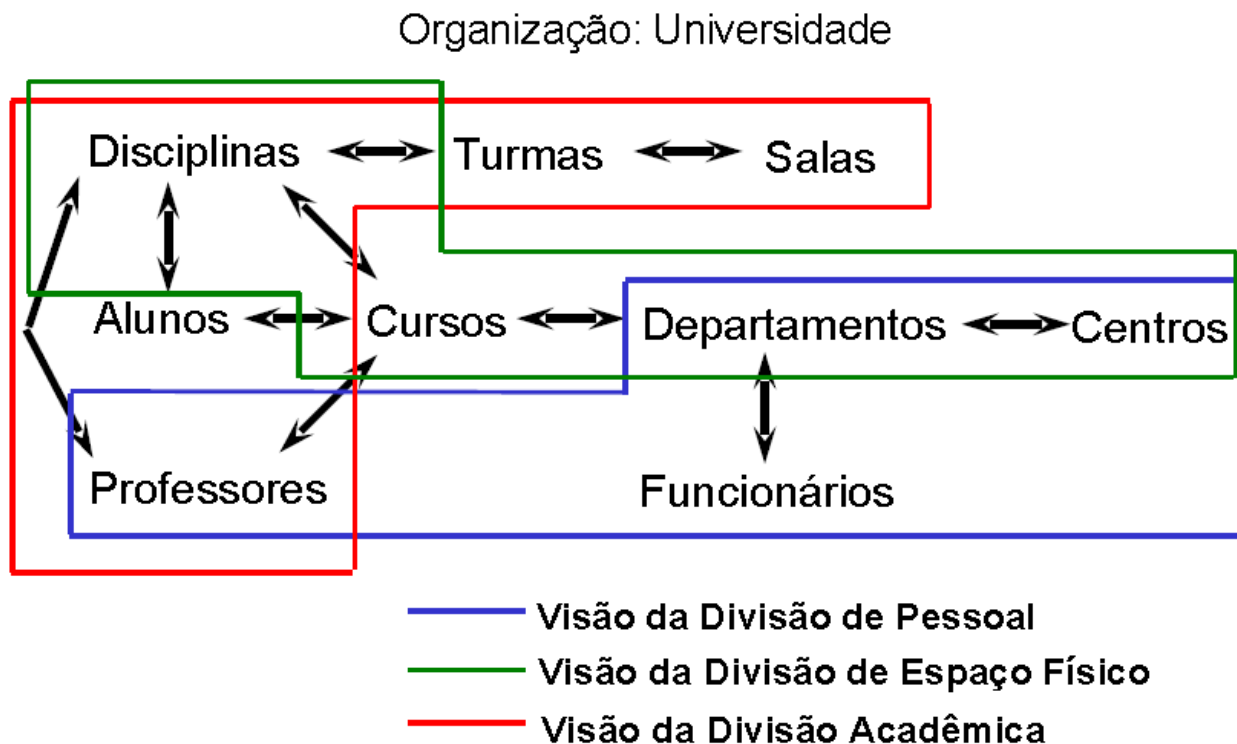
Espaço Físico

Centros
Departamentos
Cursos
Disciplinas

Pessoal

Centros
Departamentos
Professores
Funcionários

Por que usar banco de dados?



Exemplo de um banco de dados

Banco de dados

§ Banco de dados = instância de dado + meta-dados

ü Instância de dado

- Dado propriamente

ü Meta-dados

- *Dicionário de dados*
 - Esquema da base de dados
 - Acessado através de linguagens de definição de dados

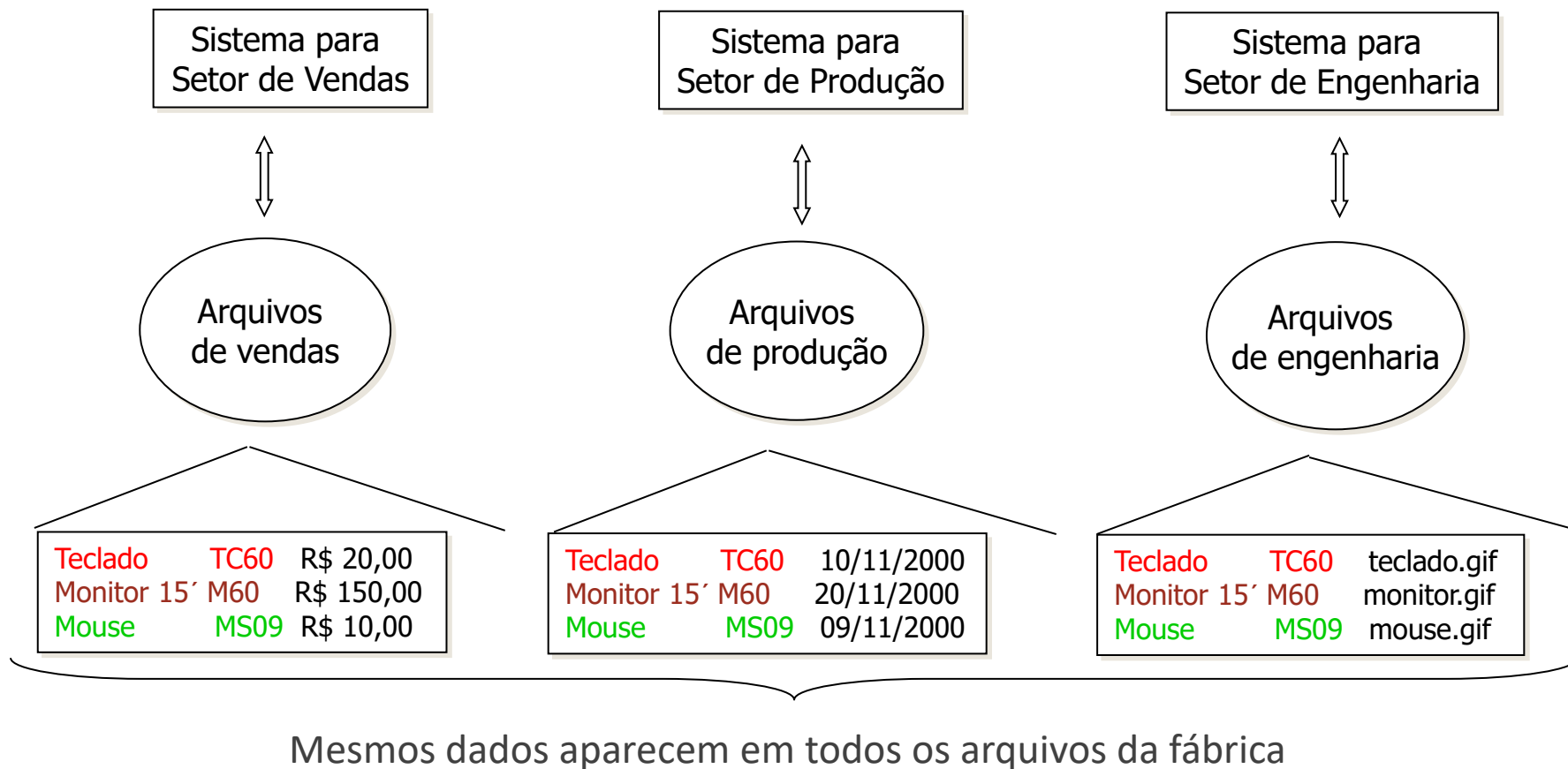
Nem sempre foi assim...

- § Sistemas de Arquivos (armazenados em pastas, no disco):
 - ü Funcionalidades oferecidas
 - Registros de tamanho fixo com campos de tipos diferentes
 - Possibilidade de memória virtual e persistência
 - Índices: *hash*, árvore-B
 - Bloqueio de arquivo e registro para concorrência
- § Dados de diferentes aplicações não estão integrados
- § Dados são projetados para atender uma aplicação específica



Sistemas de arquivos

- Em uma fábrica com os dados em sistemas de arquivos:





Sistemas de arquivos (dados não integrados)

- § Mesmo objeto da realidade é representado várias vezes na base de dados
 - ü Exemplo - teclado, monitor e mouse
 - § **Redundância não controlada** de dados
 - ü Não há gerência automática da redundância
 - ü Redundância leva a
 - *inconsistência dos dados*
 - *re-digitação de informações*
 - *dificuldade de extração de informações*
- Dados **pouco confiáveis** e de **baixa disponibilidade**

- § Concorrência
 - ü Difícil implementação
 - ü Políticas de acesso concorrente consistente são independentes de domínio
- § Tolerância a falhas
 - ü Falta de luz, erro de disco, interrupção de funcionamento, etc
 - ü Cópias? restauração do estado anterior? Consistência da base?
- § Segurança
 - ü Acesso diferenciado por tipo de usuário

Sistemas de arquivos

Sistemas de arquivos (gerenciamento dos arquivos)

§ Outros problemas:

- ü Número máximo de arquivos
- ü Tamanho de memória
- ü Limitações do tipo de arquivo, tipo de acesso
- ü Preocupações técnicas junto com problemas do domínio

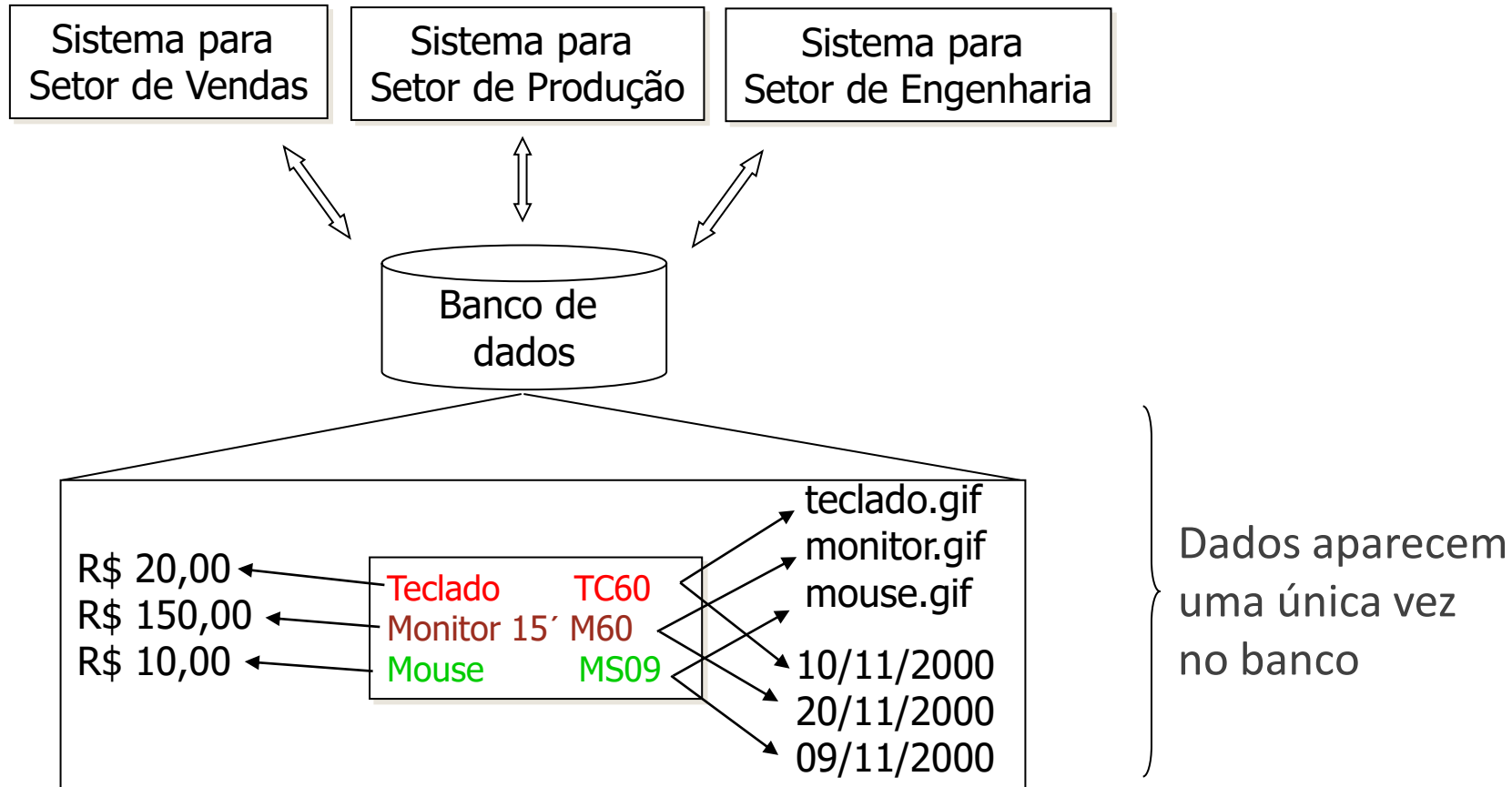
§ Exemplo: efetuar aluguel de um DVD

- ü Sem reservas? sem multas?
- ü Como registrar um empréstimo?
 - abrir arquivos (*fechando outros ...*)
 - *carregar registros na memória (abre índice, usa ponteiro, estourou memória?,)*



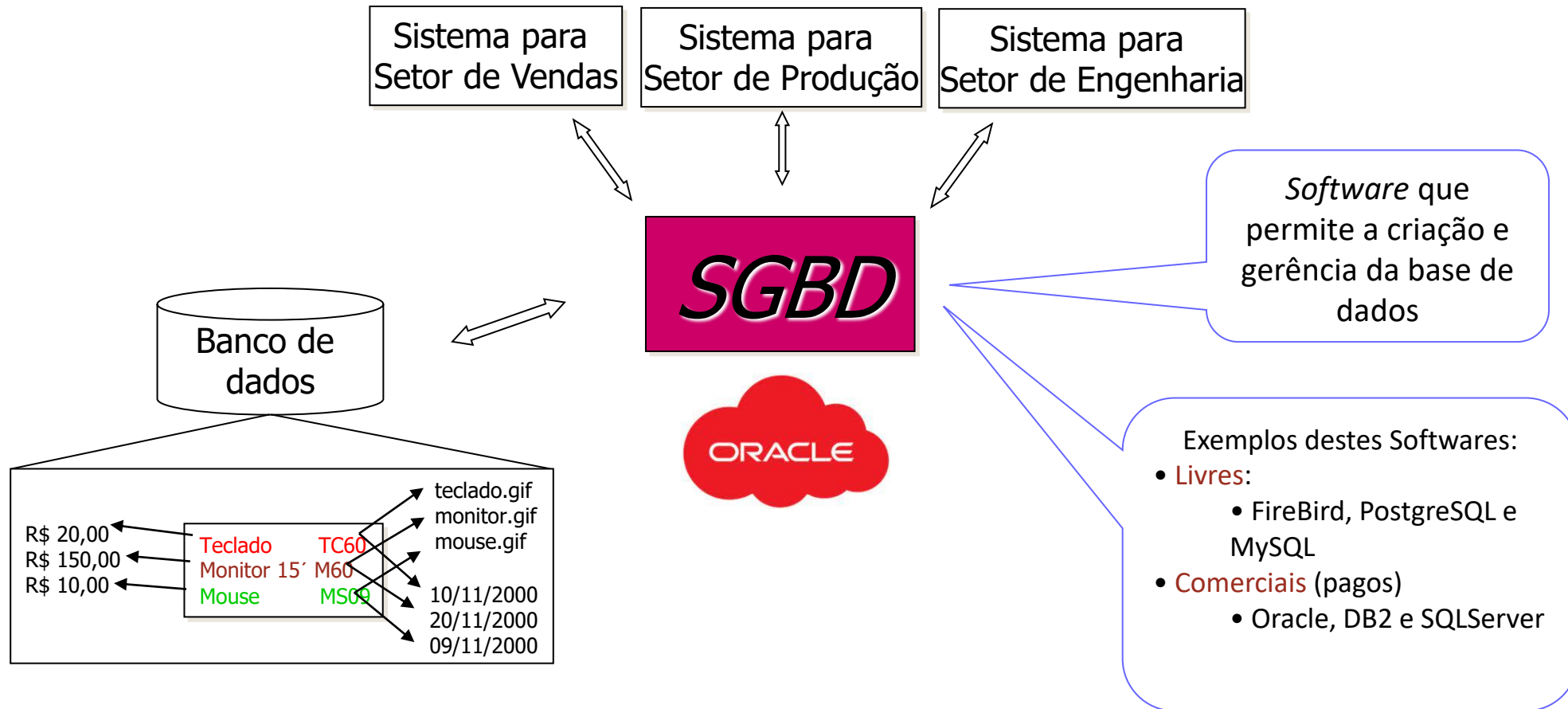
Banco de dados

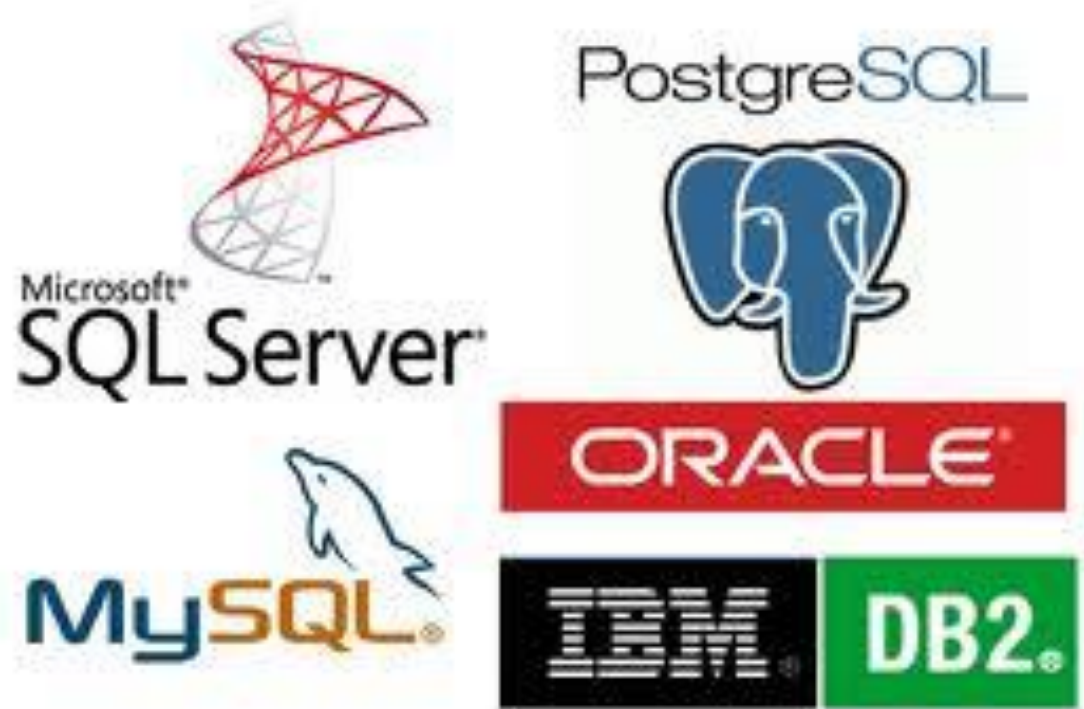
- Em uma fábrica com os dados em bancos de dados:



Gerenciamento do banco de dados

- BD de uma fábrica:





Sistema Gerenciador de Bancos de Dados (SGBD)

Um SGBD (Sistema Gerenciador de Banco de Dados) consiste em uma coleção de dados inter-relacionados **e em um conjunto de programas para acessá-los**

SGBDs são projetados para gerenciar grandes grupos de informações



SGBD

O gerenciamento envolve

- A definição de estruturas para o armazenamento da informação
- O fornecimento de mecanismos para manipular as informações

Quando vários usuários acessam os dados o SGBD precisa garantir a INTEGRIDADE dos dados, evitando resultados anômalos

Objetivos de um SGBD

Isolar os usuários dos detalhes mais internos do banco de dados (abstração de dados).

Prover independência de dados às aplicações (estrutura física de armazenamento e à estratégia de acesso).

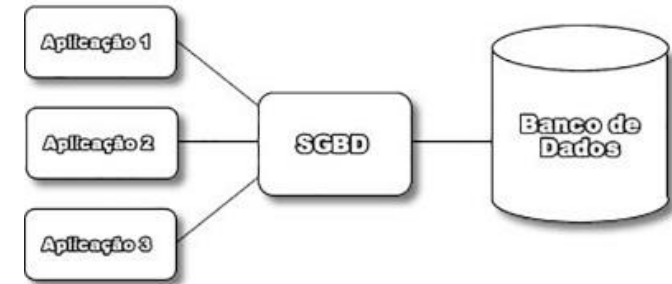
Vantagens:

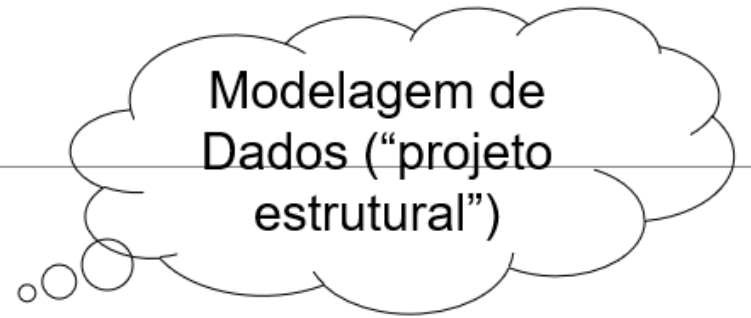
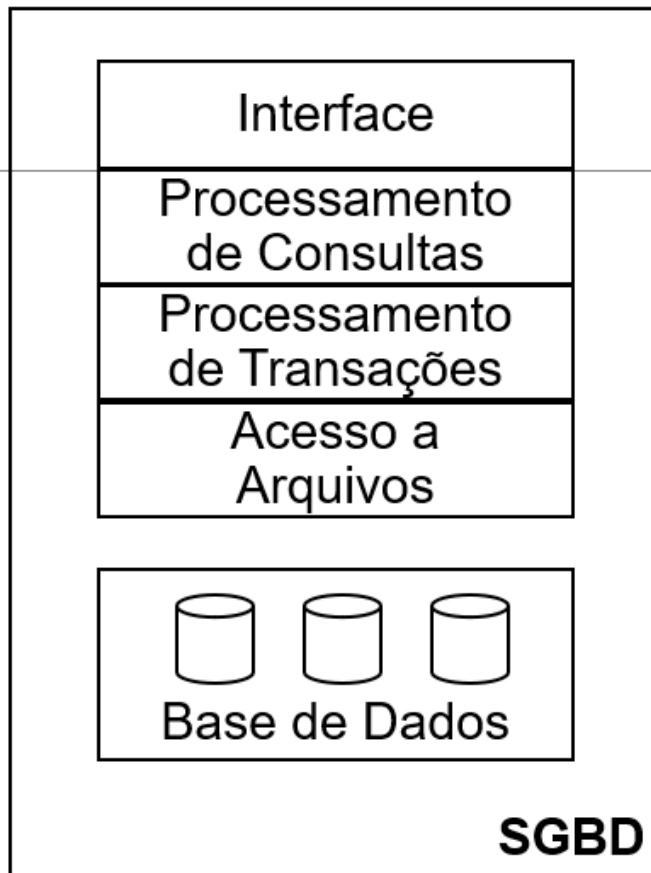
- rapidez na manipulação e no acesso à informação,
- redução do esforço humano (desenvolvimento e utilização),
- redução da redundância e da inconsistência de informações,
- redução de problemas de integridade,
- compartilhamento de dados,
- aplicação automática de restrições de segurança,
- controle integrado de informações distribuídas fisicamente.

Objetivos de um SGBD

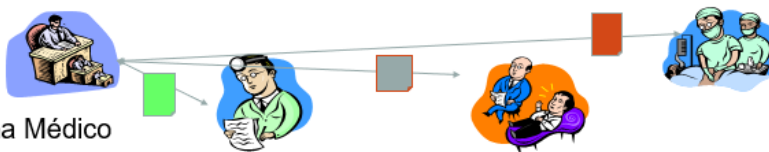
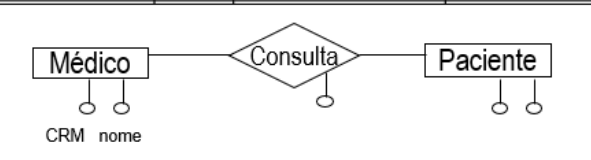
O grande objetivo de um SGBD é prover aos usuários uma visão **ABSTRATA** dos dados

- O sistema omite certos detalhes de como os dados são armazenados e mantidos
- Mas oferece mecanismos eficientes para BUSCA e ARMAZENAMENTO





Arquitetura Geral de um SGBD

Mundo Real	 <p>Sistema Médico</p>			
Modelo Conceitual (modelo abstrato dos dados)	<ul style="list-style-type: none"> ♦ Independente do modelo de dados ♦ Independente do SGBD 			
Modelo Lógico (estrutura dos dados)	<ul style="list-style-type: none"> ♦ Dependente do modelo de dados ♦ Independente do SGBD <p>Médico (CRM, Nome)</p>	Relacional	Orientado a Objetos	Objeto-relacional
Modelo Físico	<ul style="list-style-type: none"> ♦ Dependente do modelo de dados ♦ Dependente do SGBD 	<ul style="list-style-type: none"> ♦ Organização física dos dados ♦ Estruturas de armazenamento de dados ♦ Índices de acesso 		

Abstração de Dados

Modelos de Dados

CONSISTENCY
IS THE KEY!



Modelos de Dados

Um **modelo de dados** é uma coleção de ferramentas conceituais para a **descrição** de dados, **relacionamentos**, semântica de dados e restrições de **consistência**

Modelos de Dados

Modelos de Dados (conceitual)

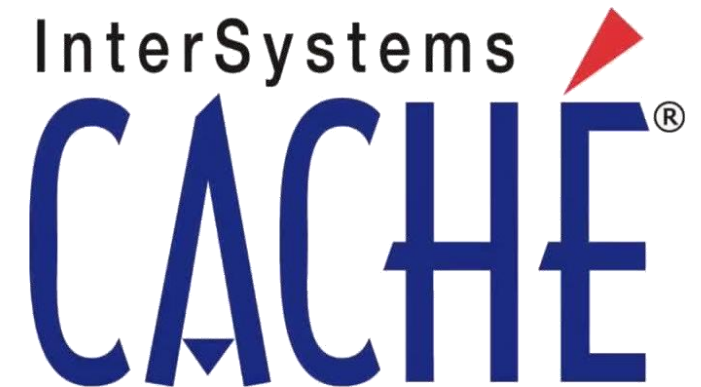
- Entidade-Relacionamento (ER)
- Orientado a Objetos (OO)

Modelos de Dados (lógicos)

- Redes
- Hierárquico
- Relacional
- Objeto-relacional
- Orientado a Objetos

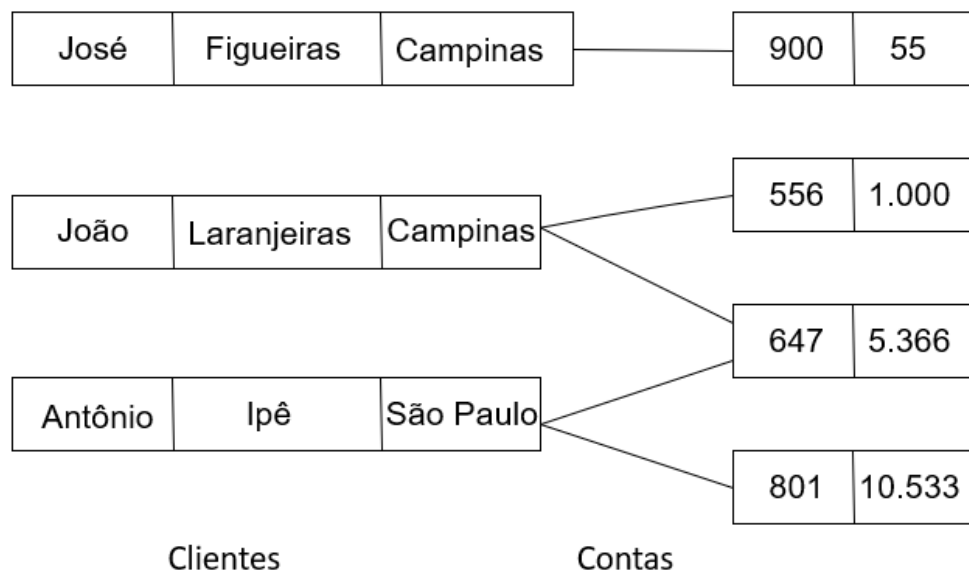


Modelos mais antigos



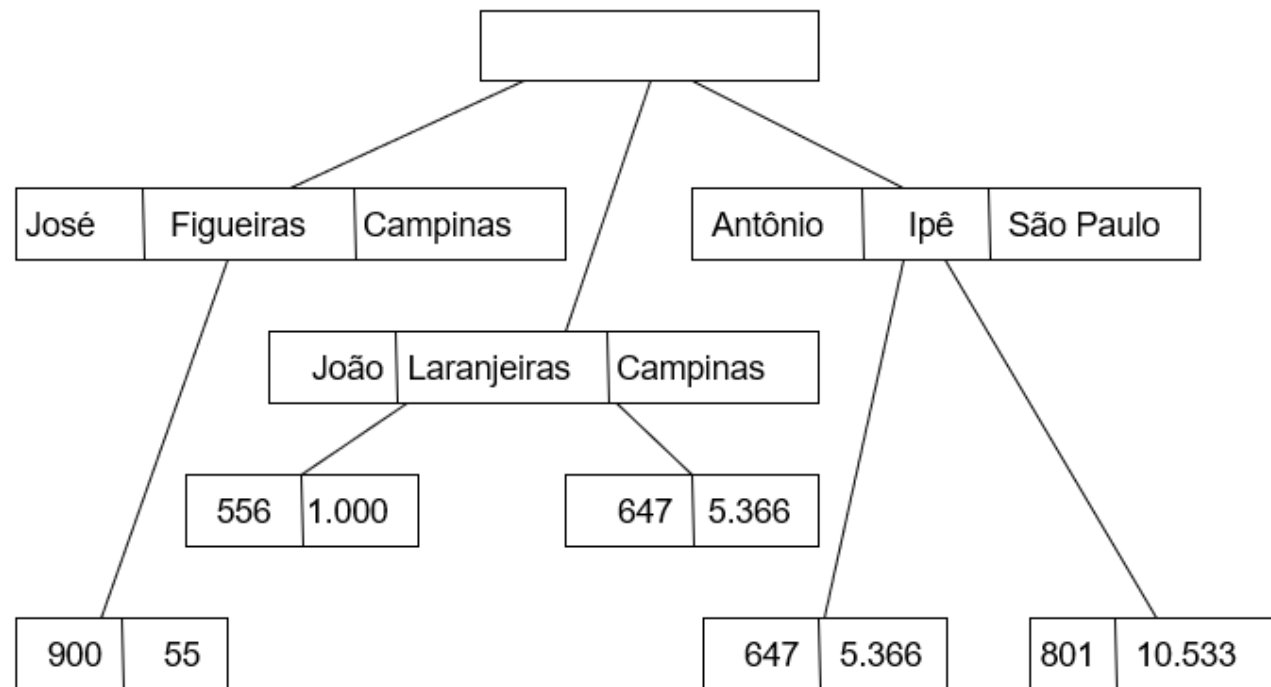
nome	rua	cidade	conta	saldo
José	Figueiras	Campinas	900	55
João	Laranjeiras	Campinas	556	1.000
João	Laranjeiras	Campinas	647	5.366
Antônio	Ipê	São Paulo	647	5.366
Antônio	Ipê	São Paulo	801	10.533

Exemplo das Informações em um
Banco de Dados



O Modelo de Redes

Os dados são representados por coleções de registros e os relacionamentos por elos



O Modelo Hierárquico

Os dados e relacionamentos são representados por registros e ligações, respectivamente.
Os registros são organizados como coleções arbitrárias de árvores.

Tabela Cliente-Conta
(relacionamento)

cód-cliente	nro-conta
015	900
021	556
021	647
037	647
037	801

Tabela Cliente (dados)

cód-cliente	nome	rua	cidade
015	José	Figueiras	Campinas
021	João	Laranjeiras	Campinas
037	Antônio	Ipê	São Paulo

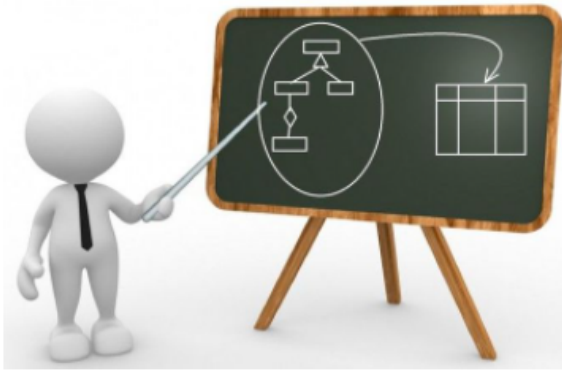
Tabela Conta (dados)

nro-conta	saldo
900	55
556	1.000
647	5.366
801	10.533

O Modelo Relacional

Tabelas (ou relações, ou entidades)

Todos os dados de um banco de dados relacional (BDR) são armazenados em tabelas. Uma tabela é uma simples estrutura de linhas e colunas. Em uma tabela, cada linha contém um mesmo de colunas. Em um banco de dados podem existir uma ou centenas de tabelas, sendo que o limite pode ser imposto tanto pela ferramenta de software utilizada, quanto pelos recursos de hardware disponíveis no equipamento.



As tabelas associam-se entre si por meio de regras de relacionamentos, que consiste em associar um ou vários atributos de uma tabela com vários atributos de outra tabela.

Exemplo: A tabela funcionário relaciona-se com a tabela cargo. Por este relacionamento, esta última tabela fornece a lista de cargos para a funcionário.

Modelo teórico usado para representar conceitualmente um BD, idealizado por Codd (1970). Baseado numa estrutura de dados simples chamada relação. É o modelo mais amplamente usado principalmente em aplicações convencionais de BD.

O Modelo Relacional

Registros (ou tuplas)

Cada linha formada por uma lista ordenada de colunas representa um registro, ou tupla. Os registros não precisam conter informações em todas as colunas, podendo assumir valores nulos quando assim se fizer necessário.

Resumidamente, um registro é uma instância de uma tabela, ou entidade. O start da modelagem se dá a partir das ENTIDADES. Uma entidade é uma representação de um conjunto de informações sobre determinado conceito do sistema. Toda entidade possui ATRIBUTOS, que são as informações que referenciam a entidade. Para exemplificar no sistema de controle de Biblioteca, partimos do conceito principal que é o empréstimo de obras por usuários da biblioteca. A partir deste conceito inicial, vamos ramificando e descobrindo novos conceitos. Podemos iniciar nosso raciocínio da seguinte forma:

"Uma biblioteca possui Obras literárias que podem ser tomadas em empréstimos pelos usuários credenciados."

O Modelo Relacional

Podemos rapidamente enxergar um cadastro de livros, um cadastro de usuários e um registro de empréstimos, certo? É essa visão que temos que ter ao modelarmos um banco, isto é, devemos detectar as informações que devemos armazenar.

Para identificar se aquele conceito pode ser uma entidade você deve apenas se perguntar: "Eu desejo armazenar quais informações sobre este conceito?" Se houver informações a serem armazenadas, você tem uma ENTIDADE.

- **Exemplificando:** Eu desejo armazenar os seguintes dados do livro: Título, Autor, Editora, Ano, Edição e Volume. Temos então a entidade Livro.

Veja: O empregado Pedro é uma instância (registro) da tabela funcionário, e a função Analista Comercial é a instância (registro) da tabela cargo. Uma associação entre estas duas tabelas criaria a seguinte instância de relacionamento: Pedro é Analista Comercial, onde o verbo ser representa uma ligação entre os registros distintos.

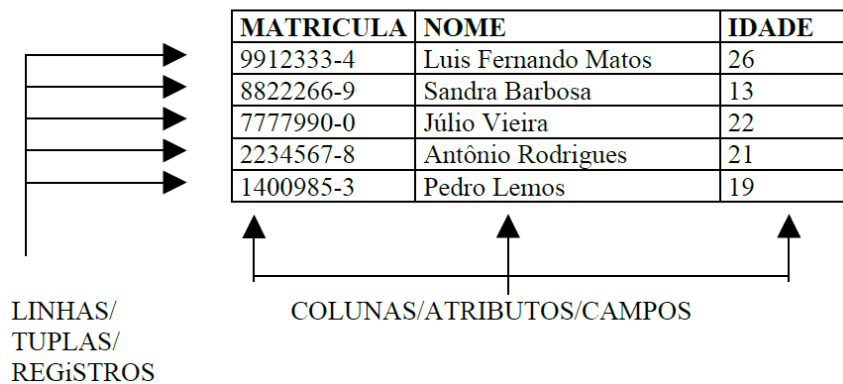
O Modelo Relacional

Modelo Relacional

Colunas (atributos)

As colunas de uma tabela são também chamadas de atributos. Ex.: O campo Nome, ou endereço de uma tabela de um BD relacional.

TABELA/RELAÇÃO/ARQUIVO



Chave

As tabelas relacionam-se umas às outras através de chaves. Uma chave é um conjunto de um ou mais atributos que determinam a unicidade de cada registro.

Por exemplo, se um banco de dados tem como chaves Código do Produto e ID Sistema, sempre que acontecer uma inserção de dados o sistema de gerenciamento de banco de dados irá fazer uma consulta para identificar se o registro já não se encontra gravado na tabela. Neste caso, um novo registro não será criado, resultando esta operação apenas da alteração do registro existente.

A unicidade dos registros, determinada por sua chave, também é fundamental para a criação dos índices.

Temos dois tipos de chaves:

Chave primária:	Chave Estrangeira:
(PK - Primary Key) é um identificador exclusivo de todas as informações de cada registro dando-lhe unicidade. A chave primária nunca se repetirá.	(FK - Foreign Key) é a chave formada através de um relacionamento com a chave primária de outra tabela. Define um relacionamento entre as tabelas e pode ocorrer repetidas vezes. Caso a chave primária seja composta na origem, a chave estrangeira também o será.

O Modelo Relacional

Diferença entre os Modelos

O modelo relacional não usa ponteiros ou ligações

O modelo relacional relaciona registros a partir de valores do registro

Funções de um SGBD

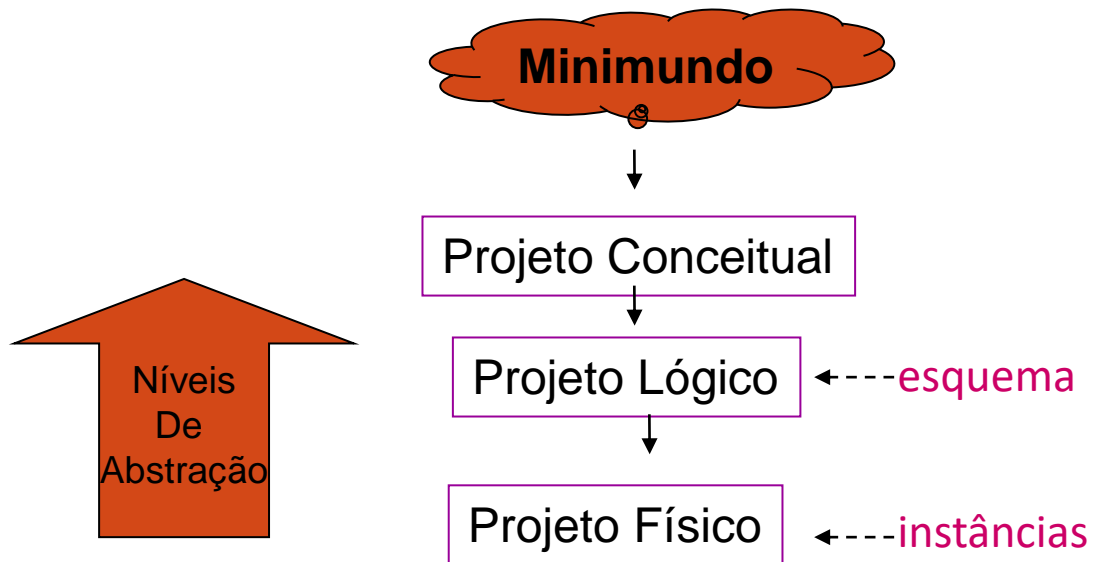
Instâncias e Esquemas

Os bancos de dados mudam a medida que informações são inseridas ou apagadas

- A coleção de informações armazenadas é chamada de **INSTÂNCIA** do banco de dados (mudam com frequência)
- O projeto geral do banco de dados é chamado **ESQUEMA** do banco de dados (não mudam com frequência)

Independência dos Dados

O uso de bancos de dados permite modificar o ESQUEMA dos dados em um nível sem afetar a definição do esquema em um nível mais alto. Isto é chamado de ***independência dos dados***



Independência dos Dados

Existem 2 tipos de Independência

- ***Independência física de dados:*** habilidade de modificar o **esquema físico** sem a necessidade de reescrever os programas aplicativos
 - Estas modificações são necessárias para melhorar o desempenho
- ***Independência lógica de dados:*** habilidade de modificar o **esquema conceitual** sem a necessidade de reescrever os programas aplicativos
 - Estas modificações são necessárias quando a estrutura lógica é alterada.
 - Exemplo: adição de um novo atributo



Independência dos Dados

A independência lógica dos dados é mais difícil de ser alcançada do que a independência física, pois os programas são bastante dependentes da estrutura lógica dos dados que eles acessam

Linguagem de Definição de Dados (DDL)

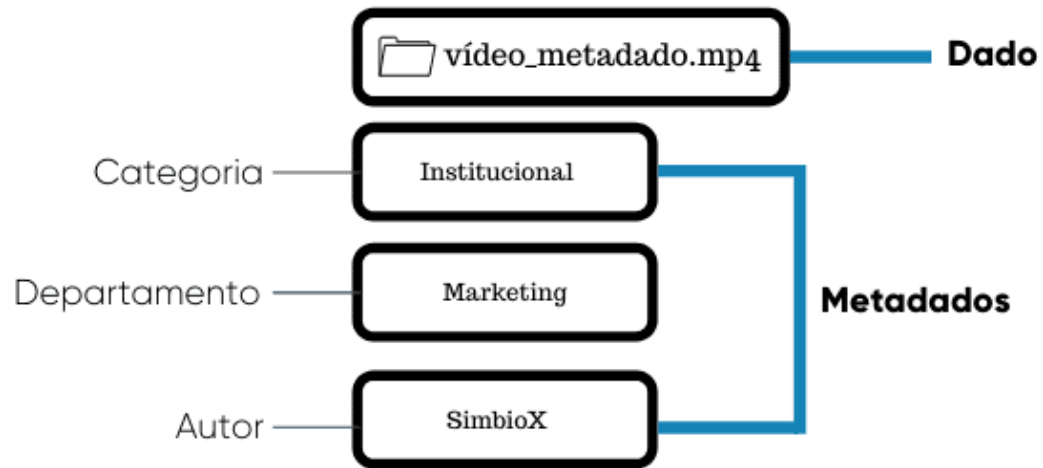
Um esquema de banco de dados é especificado por um conjunto de definições expressas por uma linguagem especial chamada ***linguagem de definição de dados (Data Definition Language)***

O resultado da compilação de comandos de uma DDL é o conjunto de tabelas que serão armazenadas no dicionário (ou diretório) de dados

Linguagem de Definição de Dados (DDL)

Um dicionário de dados contém metadados, i.e., dados sobre os dados

Este dicionário (diretório) é consultado antes que os dados sejam lidos ou modificados no sistema de banco de dados



Linguagem de Manipulação de Dados (DML)

Manipulação de dados significa:

- A busca da informação armazenada no BD
- A inserção de novas informações no BD
- A eliminação de informações do BD
- A modificação dos dados armazenados no BD

No nível físico precisamos definir algoritmos que permitam acesso eficiente aos dados

Linguagem de Manipulação de Dados (DML)

A linguagem de manipulação dos dados permite ao usuário manipular os dados da seguinte forma:

- Procedural: o usuário informa qual dado deseja acessar e como obtê-lo
- Não-procedural: o usuário informa qual dado deseja acessar SEM especificar como obtê-lo

Linguagem de Manipulação de Dados (DML)

Linguagens não-procedurais são usualmente mais fáceis de aprender e usar do que DMLs procedurais

Se o usuário NÃO especificar COMO obter os dados, as linguagens não-procedurais poderão gerar um código não tão eficiente.





Linguagem de Manipulação de Dados (DML)

Uma consulta (QUERY) é um comando de busca de uma informação no BD

A parte da DML que busca informações é chamada ***LINGUAGEM DE CONSULTA***

Usuários do Banco de Dados

Programadores de Aplicativos:

- São os usuários que escrevem os programas de aplicação através da DML
 - Exemplos de um sistema bancário são programas que geram cheques, fazem débitos e créditos em contas, transferem fundos entre contas

Usuários de alto nível

- Interagem com o sistema sem escrever programas
- Formulam consultas em uma linguagem de consulta, e cada consulta é submetida a um processador de consulta, cuja função é gerar um comando da DML

Usuários do Banco de Dados

Usuários especializados (especialistas)

- Escrevem aplicativos especializados como sistemas especialistas

Usuários ingênuos

- Interagem com o sistema invocando os programas aplicativos
 - Exemplo: um cliente do banco invocaria um programa para efetuar a transferência de 50 reais da conta A para a conta B

Usuários do Banco de Dados



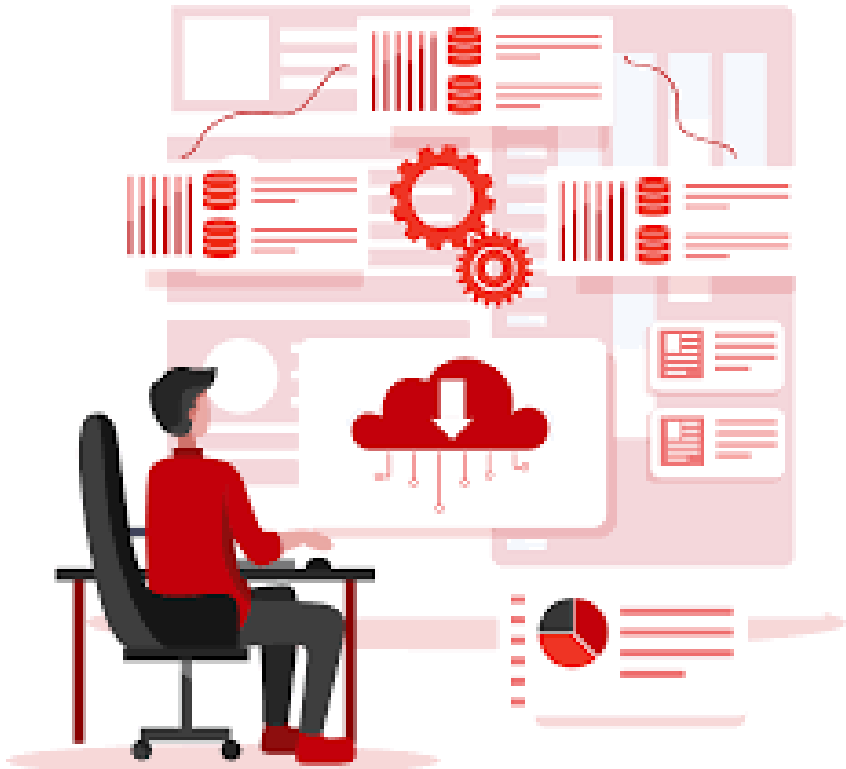
Database administrator

Administrador do banco de dados: tem o controle central dos dados e dos programas de acesso aos dados

Usuários do Banco de Dados

Funções do Administrador do banco de dados:

- Definição do esquema
- Definição de estruturas de armazenamento e métodos de acesso
- Modificação de esquema e de organização física
- Concessão de autorização para acesso aos dados
- Especificação de restrições de integridade





Gerenciador de arquivos



Gerenciador do banco de dados



Processador de consultas



Pré-compilador da DML



Compilador da DDL

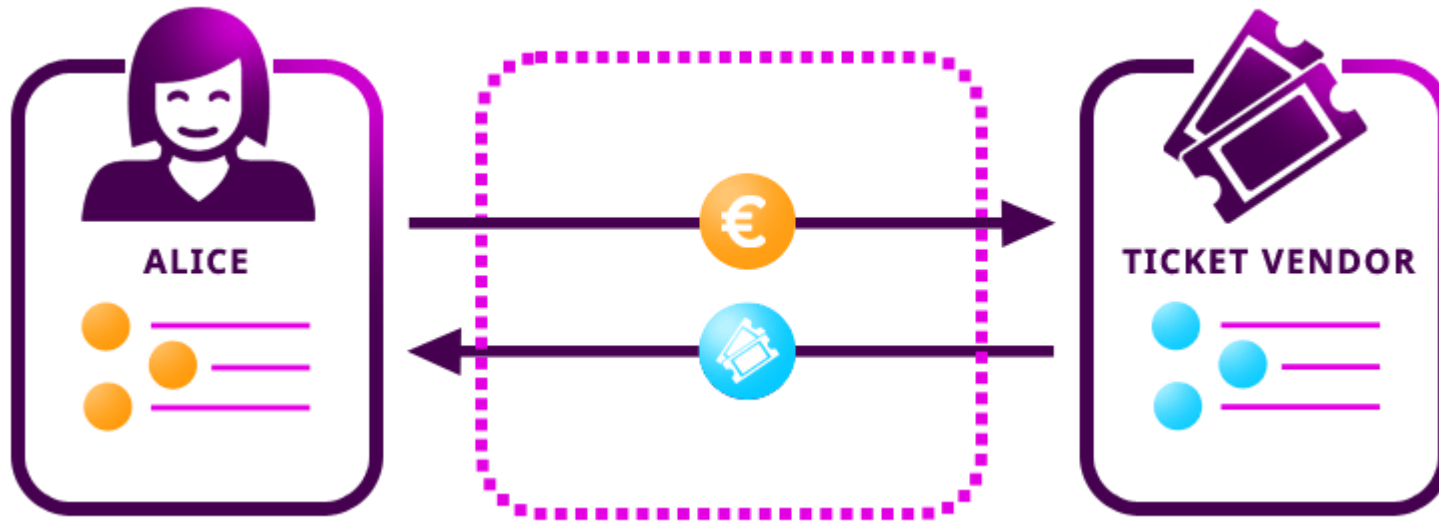
Arquivos de dados
Dicionário de dados
Índices

Estrutura Geral do Sistema

Transações

Utilizadas para controlar a integridade dos dados no Banco de dados

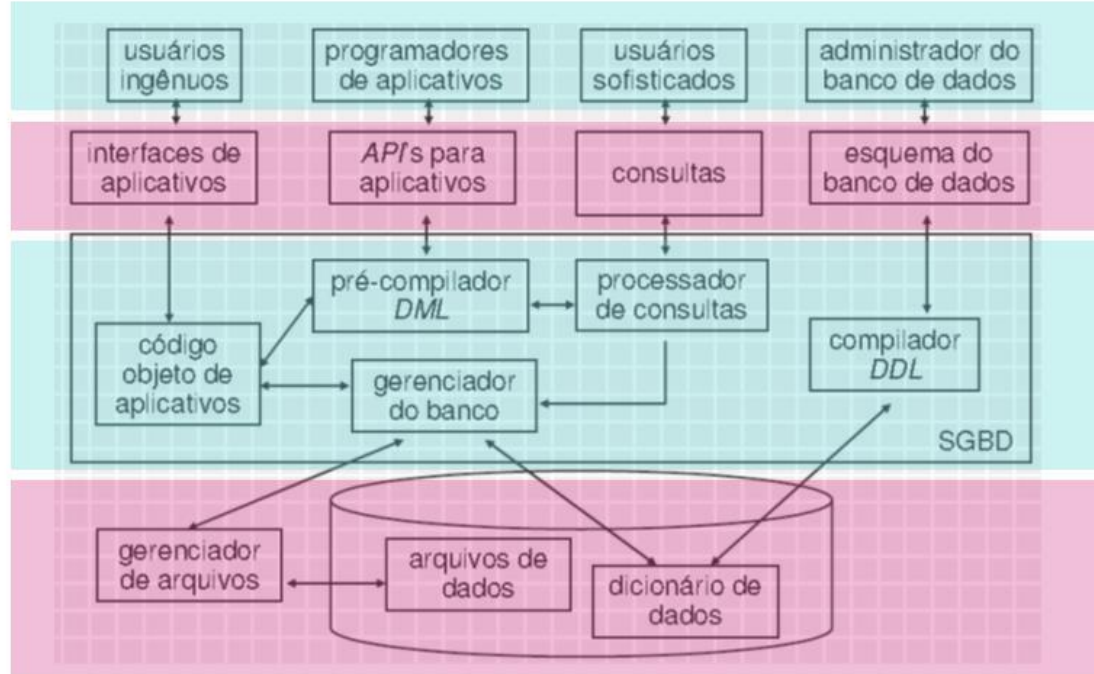
- Acessos simultâneos vários usuários
- Falhas no sistema





Otimizador de Consultas

ESCOLHE A FORMA MAIS EFICIENTE PARA EXECUÇÃO DE UMA CONSULTA



Arquitetura Geral de um SGBD

SQL – Structured Query Language



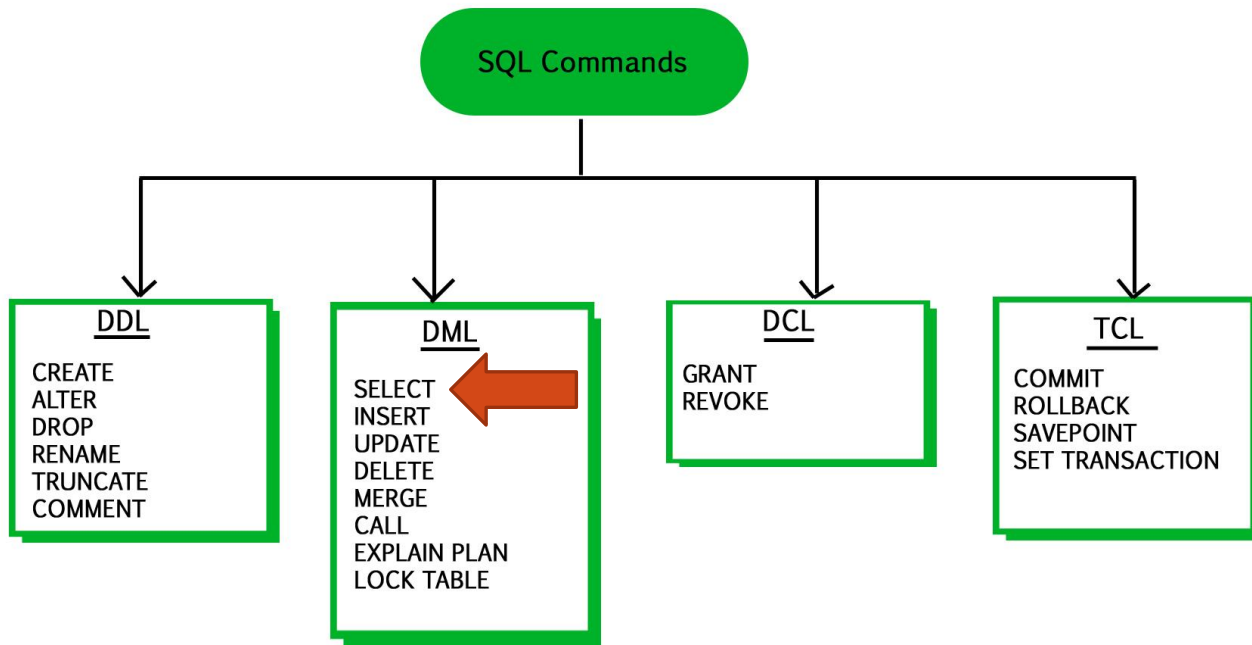
A Linguagem SQL

- A linguagem SQL é a linguagem de pesquisa declarativa padrão para banco de dados.
- Essa linguagem permite a criação de comandos não apenas para consulta de informações armazenadas no banco de dados, mas também para alterar, excluir ou mesmo modificar a estrutura do esquema de armazenamento das informações.
- O SQL é um grande padrão de banco de dados.
- Isto decorre da sua simplicidade e facilidade de uso.

A Linguagem SQL

```
1 * select * from imovel;
2 * select * from corretor;
3 * select * from aluguel;
4 * select * from proprietario;
5 * select * from inquilino;
6
7 * select * from aluguel;
8
9 * select idAluguel, valorAluguel, 12*valorAluguel as valorAnual
10 from aluguel;
11
12 * select idAluguel, valorAluguel, valorAluguel - 50 as desconto
13 from aluguel;
14
15 * select idAluguel, dtAluguel from aluguel
16 where dtAluguel between '2013-03-01' and '2013-03-31';
17
18 * select * from aluguel
19 where valorAluguel between 300 and 600;
20
21 * select * from aluguel
22 where imovel_proprietario_cpfProprietario = 03456734523 and valorAluguel <
23
24 * select * from aluguel
25 where imovel_proprietario_cpfProprietario = 03456734523 or valorAluguel <
```

- Ela se diferencia de outras linguagens de consulta a banco de dados no sentido em que uma consulta SQL especifica a forma do resultado e não o caminho para chegar a ele.
- O SQL é uma linguagem declarativa em oposição a outras linguagens procedurais. Isto reduz o ciclo de aprendizado daqueles que se iniciam na linguagem.



Comandos SQL

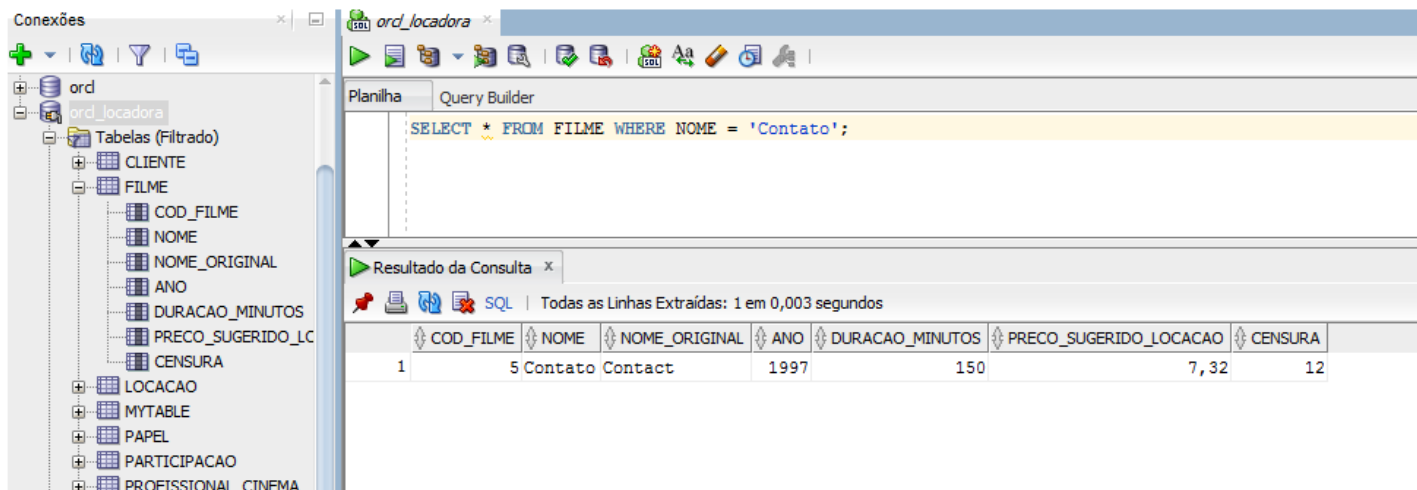
SELECT



SELECT

```
stepp@webster:~  
[stepp@webster ~] 0 $ mysql -u stepp -p  
Enter password:  
Welcome to the MySQL monitor.  Commands end with ; or \g.  
mysql> use simpsons;  
Database changed  
mysql> select * from students;  
+-----+-----+-----+-----+  
| id | name | email | password |  
+-----+-----+-----+-----+  
| 123 | Bart | bart@fox.com | bartman |  
| 404 | Ralph | ralph@fox.com | catfood |  
| 456 | Milhouse | milhouse@fox.com | fallout |  
| 888 | Lisa | lisa@fox.com | vegan |  
+-----+-----+-----+-----+  
4 rows in set (0.00 sec)
```

- O comando SELECT é composto dos atributos que desejamos, a ou as tabela(s) que possuem esses atributos e as condições que podem ajudar a filtrar os resultados desejados.
- Não é uma boa prática usar o * ou star para trazer os registros de uma tabela.
 - Procure especificar somente os campos necessários.
 - Isso ajuda o motor de execução de consultas a construir bons planos de execução.
- Se você conhecer a estrutura da tabela e seus índices, procure tirar proveito disso usando campos chaves, ou buscando e filtrando por atributos que fazem parte de chaves e índices no banco de dados.
- Uma seleção representa um filtro de algumas linhas de uma dada relação.



FROM

O Comando FROM indica a origem dos dados que queremos.

- É possível especificar mais de uma tabela no comando FROM, porém, se você indicar mais de uma tabela no comando FROM, lembre-se de indicar os campos que fazem o relacionamento entre as tabelas mencionadas na cláusula FROM.
- Perceba que a função da cláusula WHERE combinada com a condição de seleção utilizada permite que somente sejam mostrados os registros em que o conteúdo do campo NOME seja Contato.

ord_locadora

Planilha Query Builder

```
SELECT * FROM FILME WHERE NOME = 'Contato';
SELECT * FROM CLIENTE WHERE UF = 'RJ' OR UF = 'SP';
```

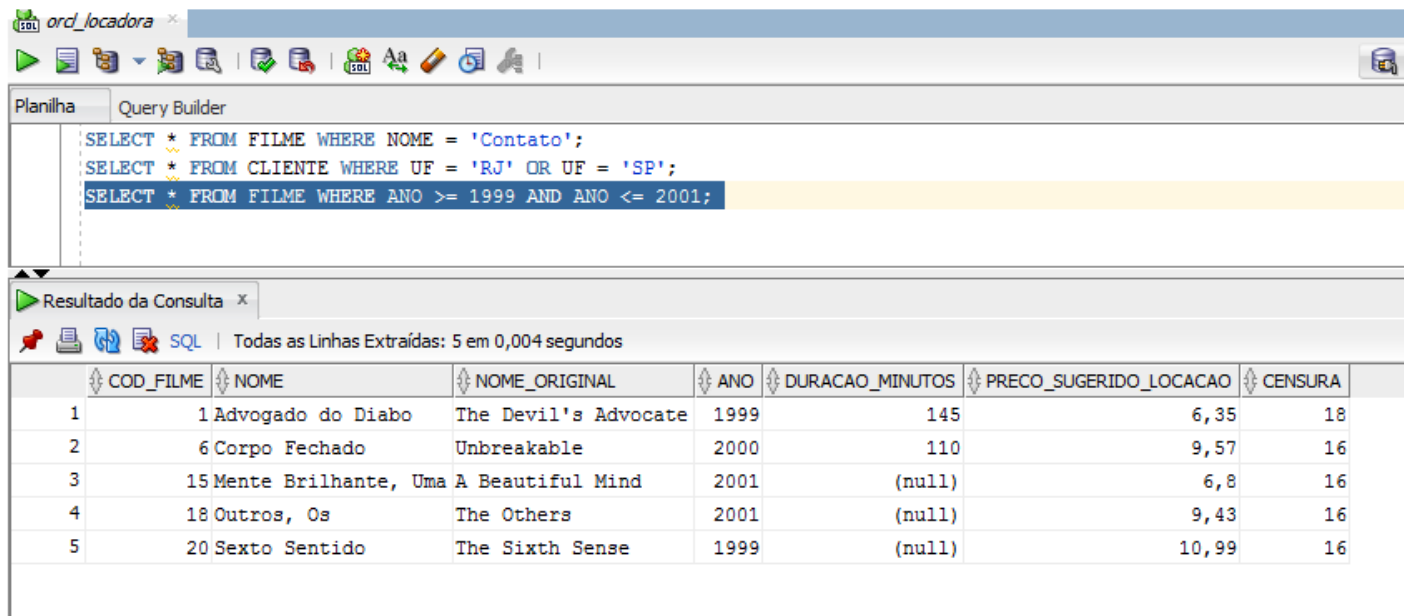
Resultado da Consulta x

Todas as Linhas Extraídas: 6 em 0,278 segundos

	COD_CLIENTE	CLIENTE_TITULAR	NOME	TELEFONE	ENDERECO	CIDADE	UF
1	6	(null)	Jose Bento Renato Monteiro Lobato	(null)	Rua Taubaté, 1882	Guaratinguetá	SP
2	8	(null)	Graciliano Ramos de Oliveira	(null)	Rua Qebrangulo, 1892	Queluz	SP
3	12	(null)	Bernardo Élis Fleury de Campos Curado	(null)	Rua Corumbá de Goiás, 1915/201	Três Rios	RJ
4	16	(null)	Samuel Rawet	(null)	Av, Polônia, 1929	Petrópolis	RJ
5	21	(null)	José J, Veiga	(null)	Rua Corumbá de Goiás, 1915/301	Três Rios	RJ
6	23		16Clarice Lispector	(null)	Av, Ucrânia, 1920	Petrópolis	RJ

OR

- O segundo exemplo traz um filtro utilizando o operador lógico OR.
- Perceba que estamos trazendo aqueles clientes de São Paulo MAIS os clientes do Rio de Janeiro.



The screenshot shows the 'ord_locadora' application interface. The 'Query Builder' tab is active, displaying three SQL queries. The third query, 'SELECT * FROM FILME WHERE ANO >= 1999 AND ANO <= 2001;', is highlighted. Below the queries, the 'Resultado da Consulta' tab shows a table with 5 rows of data. The table has 7 columns: COD_FILME, NOME, NOME_ORIGINAL, ANO, DURACAO_MINUTOS, PRECO_SUGERIDO_LOCACAO, and CENSURA.

	COD_FILME	NOME	NOME_ORIGINAL	ANO	DURACAO_MINUTOS	PRECO_SUGERIDO_LOCACAO	CENSURA
1	1	Advogado do Diabo	The Devil's Advocate	1999	145	6,35	18
2	6	Corpo Fechado	Unbreakable	2000	110	9,57	16
3	15	Mente Brilhante, Uma	A Beautiful Mind	2001	(null)	6,8	16
4	18	Outros, Os	The Others	2001	(null)	9,43	16
5	20	Sexto Sentido	The Sixth Sense	1999	(null)	10,99	16

AND

- Vamos ao terceiro exemplo, agora utilizando mais um operador lógico AND.
- Perceba que, enquanto OR tinha um caráter aditivo, a cláusula AND restringe o resultado. No caso, para que o filme seja mostrado devem ser obedecidas as DUAS condições:
 - O ano de produção deve ser 1999 ou posterior; E
 - O ano de produção deve ser 2001 ou anterior.

ord_locadora

Planilha Query Builder

SELECT DESCRICAO FROM PAPEL;

Resultado da Consulta

Todas as Linhas Extraídas: 5 em 0,007 segundos

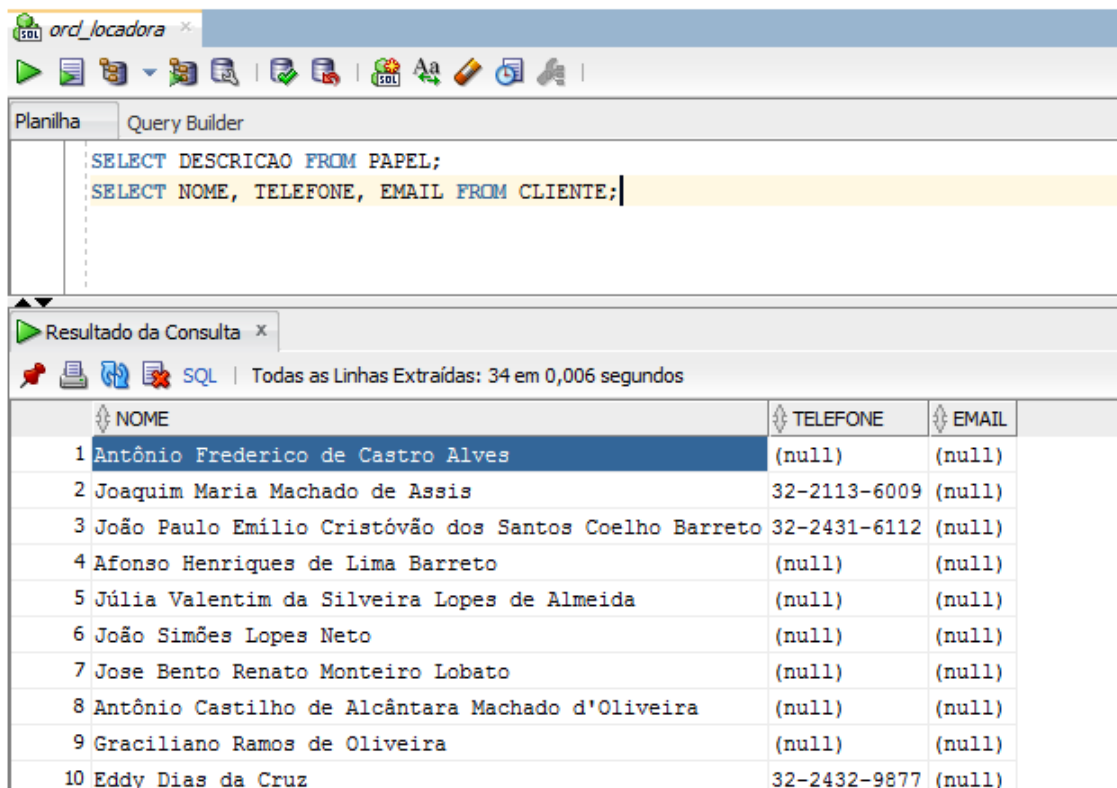
	DESCRICAO
1	ATOR COADJUVANTE
2	ATOR PRINCIPAL
3	ATRIZ COADJUVANTE
4	ATRIZ PRINCIPAL
5	DIRETOR

Projeção

- Uma projeção representa um filtro de algumas colunas de uma dada relação.
- No primeiro exemplo, filtramos na tabela PAPEL apenas o campo Descrição.
- Note que, caso fosse desejado apresentar o conteúdo de todas as colunas deveria ser usado "*" .

Projeção

- O segundo exemplo exhibe apenas as colunas NOME, TELEFONE e EMAIL de clientes.
- Observe que muitos telefones e, quem sabe, todos os e-mails, estão vazios, ou seja, nenhum valor foi informado para esses campos.



The screenshot shows a SQL query builder window titled 'ord_locadora'. The 'Query Builder' tab is active, displaying the following SQL query:

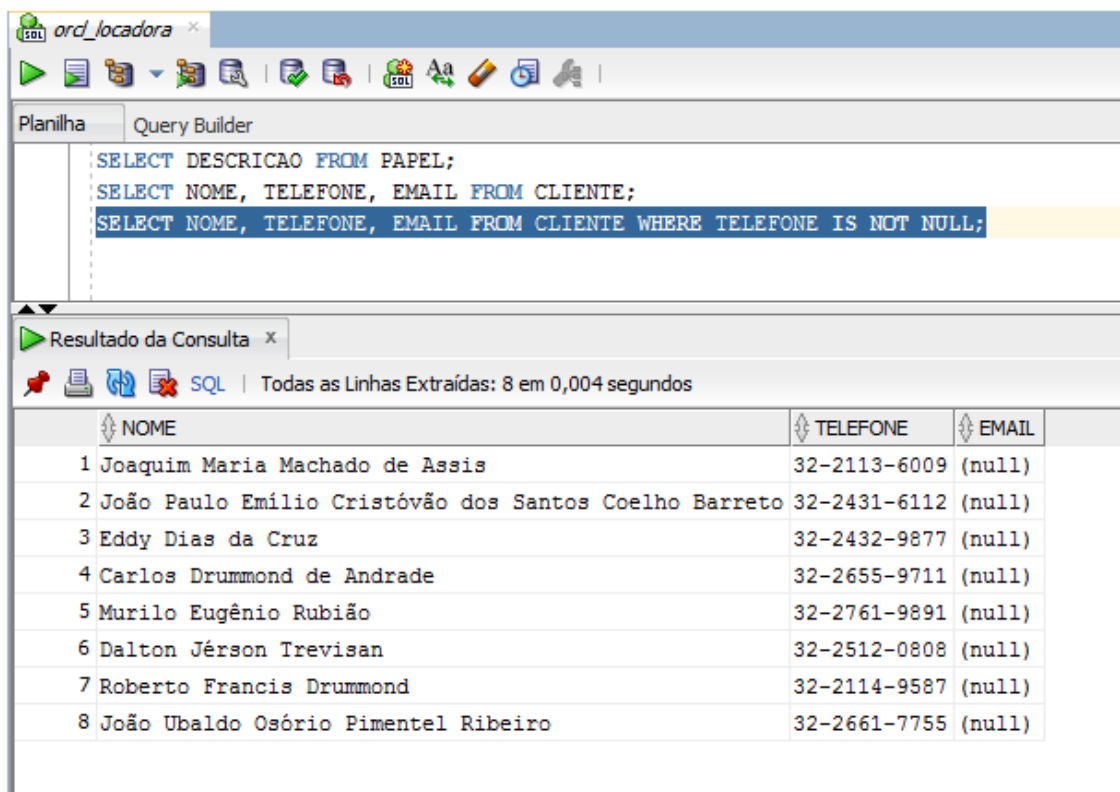
```
SELECT DESCRICAO FROM PAPEL;  
SELECT NOME, TELEFONE, EMAIL FROM CLIENTE;
```

Below the query editor, the 'Resultado da Consulta' (Query Result) window is open, showing the results of the second query. The status bar indicates 'Todas as Linhas Extraídas: 34 em 0,006 segundos' (All rows extracted: 34 in 0.006 seconds). The results are displayed in a table with three columns: NOME, TELEFONE, and EMAIL.

	NOME	TELEFONE	EMAIL
1	Antônio Frederico de Castro Alves	(null)	(null)
2	Joaquim Maria Machado de Assis	32-2113-6009	(null)
3	João Paulo Emilio Cristóvão dos Santos Coelho Barreto	32-2431-6112	(null)
4	Afonso Henriques de Lima Barreto	(null)	(null)
5	Júlia Valentim da Silveira Lopes de Almeida	(null)	(null)
6	João Simões Lopes Neto	(null)	(null)
7	Jose Bento Renato Monteiro Lobato	(null)	(null)
8	Antônio Castilho de Alcântara Machado d'Oliveira	(null)	(null)
9	Graciliano Ramos de Oliveira	(null)	(null)
10	Eddy Dias da Cruz	32-2432-9877	(null)

Projeção

- Poderíamos então, por exemplo, conjugar seleção e projeção para mostrar somente os clientes que efetivamente possuam telefone informado.
- Valores nulos possuem tratamento especial (veja como utilizamos IS NULL).



The screenshot shows a SQL query builder window titled 'ord_locadora'. The 'Query Builder' tab is active, displaying the following SQL query:

```
SELECT DESCRICAO FROM PAPEL;  
SELECT NOME, TELEFONE, EMAIL FROM CLIENTE;  
SELECT NOME, TELEFONE, EMAIL FROM CLIENTE WHERE TELEFONE IS NOT NULL;
```

The 'Resultado da Consulta' tab is also visible, showing the results of the query. The results are displayed in a table with 8 rows and 3 columns: NOME, TELEFONE, and EMAIL. The EMAIL column contains null values for all rows.

	NOME	TELEFONE	EMAIL
1	Joaquim Maria Machado de Assis	32-2113-6009	(null)
2	João Paulo Emílio Cristóvão dos Santos Coelho Barreto	32-2431-6112	(null)
3	Eddy Dias da Cruz	32-2432-9877	(null)
4	Carlos Drummond de Andrade	32-2655-9711	(null)
5	Murilo Eugênio Rubião	32-2761-9891	(null)
6	Dalton Jêrson Trevisan	32-2512-0808	(null)
7	Roberto Francis Drummond	32-2114-9587	(null)
8	João Ubaldo Osório Pimentel Ribeiro	32-2661-7755	(null)

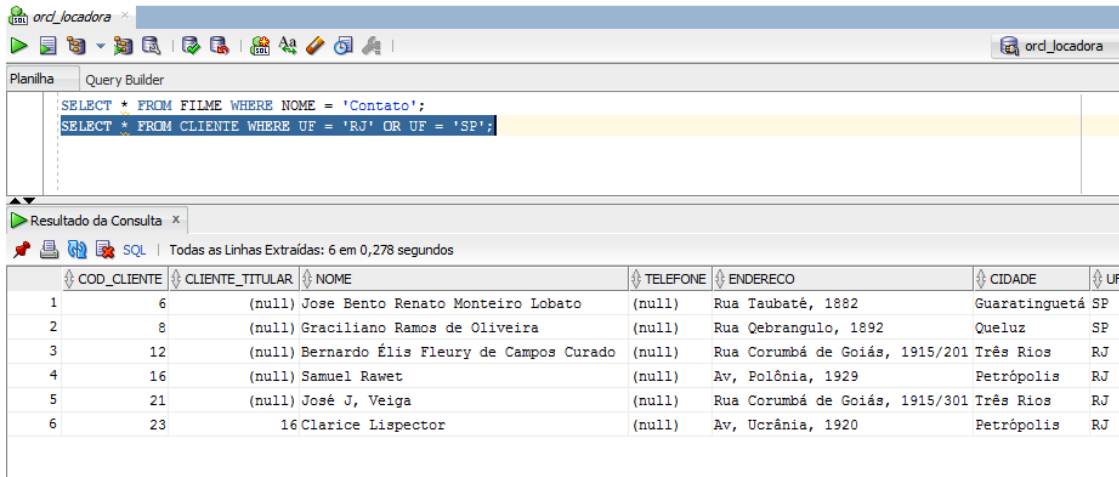
UNION

```
SELECT * FROM CLIENTE WHERE UF = 'RJ'
```

UNION

```
SELECT * FROM CLIENTE WHERE UF = 'SP';
```

Estamos criando uma união entre duas relações: a compreendida pela seleção de clientes residentes no Rio de Janeiro com aquela formada por clientes residentes em São Paulo.



The screenshot shows a database query tool interface. The top part is the 'Query Builder' window, which contains the following SQL query:

```
SELECT * FROM FILME WHERE NOME = 'Contato';  
SELECT * FROM CLIENTE WHERE UF = 'RJ' OR UF = 'SP';
```

Below the query builder is the 'Resultado da Consulta' (Query Result) window, which displays the results of the second query. The results are shown in a table with the following columns: COD_CLIENTE, CLIENTE_TITULAR, NOME, TELEFONE, ENDEREÇO, CIDADE, and UF. The table contains 6 rows of data.

	COD_CLIENTE	CLIENTE_TITULAR	NOME	TELEFONE	ENDEREÇO	CIDADE	UF
1	6	(null)	Jose Bento Renato Monteiro Lobato	(null)	Rua Taubaté, 1882	Guaratinguetá	SP
2	8	(null)	Graciliano Ramos de Oliveira	(null)	Rua Qebrangulo, 1892	Queluz	SP
3	12	(null)	Bernardo Élis Fleury de Campos Curado	(null)	Rua Corumbá de Goiás, 1915/201	Três Rios	RJ
4	16	(null)	Samuel Rawet	(null)	Av, Polônia, 1929	Petrópolis	RJ
5	21	(null)	José J, Veiga	(null)	Rua Corumbá de Goiás, 1915/301	Três Rios	RJ
6	23		16 Clarice Lispector	(null)	Av, Ucrânia, 1920	Petrópolis	RJ

ord_locadora

Planilha Query Builder

```
select * from
profissional_cinema cross join papel;
```

Resultado da Consulta

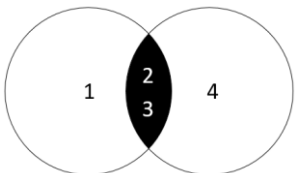
Todas as Linhas Extraídas: 325 em 0,142 segundos

	COD_PROFISSIONAL_CINEMA	NOME	DT_NASCIMENTO	PAIS_ORIGEM	COD_PAPEL	DESCRICAO
308	48	Nicole Kidman	20/06/67	EUA	5	DIRETOR
309	49	Patrick Swayze	18/08/52	EUA	5	DIRETOR
310	50	Robert Downey Jr,	04/04/65	EUA	5	DIRETOR
311	51	Robert Zemeckis	14/05/51	EUA	5	DIRETOR
312	52	Robin Williams	21/07/51	EUA	5	DIRETOR
313	53	Robin Wright Penn	08/04/66	EUA	5	DIRETOR
314	54	Russell Crowe	07/04/64	Nova Zelândia	5	DIRETOR
315	55	Samuel L. Jackson	21/12/48	EUA	5	DIRETOR
316	56	Sean Patrick Fla...	11/10/65	EUA	5	DIRETOR
317	57	Susan Tyrrell	18/03/45	EUA	5	DIRETOR
318	58	Taylor Hackford	31/12/44	EUA	5	DIRETOR
319	59	Teresa Wright	27/10/18	EUA	5	DIRETOR
320	60	Tony Goldwyn	20/05/60	EUA	5	DIRETOR
321	61	Timothy Hutton	16/08/60	EUA	5	DIRETOR
322	62	Victor Salva	29/03/58	EUA	5	DIRETOR
323	63	Vincent Ward	(null)	Nova Zelândia	5	DIRETOR
324	64	Warren Beatty	30/03/37	EUA	5	DIRETOR
325	65	Whoopi Goldberg	13/11/55	EUA	5	DIRETOR

Produto Cartesiano

Um Produto Cartesiano entre duas tabelas resulta em uma relação contendo o produto das quantidades de linhas e a soma das quantidades de colunas das tabelas envolvidas.

- Por exemplo, sabemos que a tabela de papéis possui 5 linhas e 2 colunas; a tabela de profissionais de cinema tem 65 linhas e 4 colunas.
- Caso quiséssemos imaginar como seria uma combinação onde todos os profissionais pudessem estar associados a todos os papéis, teríamos uma absurda relação com 6 colunas (2 + 4) e 325 (5 x 65) linhas!



ord_locadora

Planilha Query Builder

```
SELECT NOME, TELEFONE, UF FROM CLIENTE WHERE UF = 'MG'  
INTERSECT  
SELECT NOME, TELEFONE, UF FROM CLIENTE WHERE TELEFONE IS NOT NULL;
```

Resultado da Consulta

Todas as Linhas Extraídas: 8 em 0,002 segundos

	NOME	TELEFONE	UF
1	Carlos Drummond de Andrade	32-2655-9711	MG
2	Dalton Jerson Trevisan	32-2512-0808	MG
3	Eddy Dias da Cruz	32-2432-9877	MG
4	Joaquim Maria Machado de Assis	32-2113-6009	MG
5	João Paulo Emilio Cristóvão dos Santos Coelho Barreto	32-2431-6112	MG
6	João Ubaldo Osório Pimentel Ribeiro	32-2661-7755	MG
7	Murilo Eugênio Rubião	32-2761-9891	MG
8	Roberto Francis Drummond	32-2114-9587	MG

Interseção

Uma Interseção representa um conjunto de linhas comuns a duas relações.

- Por exemplo, caso quiséssemos descobrir a interseção entre os clientes que moram em Minas Gerais com aqueles possuindo telefone
- Resultado igual a 8 registros

ord_locadora x ord_locadora~1 x

Planilha Query Builder

```
SELECT COD_FILME
FROM PROFISSIONAL_CINEMA PC INNER JOIN PARTICIPACAO P
ON PC.COD_PROFISSIONAL_CINEMA = P.COD_PROFISSIONAL_CINEMA
WHERE PC.NOME = 'Bruce Willis';
```

Resultado da Consulta x

Todas as Linhas Extraídas: 2 em 0,005 segundos

COD_FILME	
1	6
2	20

Junção

- Uma junção permite reunir dados armazenados em tabelas diferentes.
- Por exemplo, suponha que desejássemos mostrar os códigos dos filmes nos quais Bruce Willis participou

ord_locadora x ord_locadora~1 x

Planilha Query Builder

```
SELECT COD_FILME
FROM PROFISSIONAL_CINEMA PC INNER JOIN PARTICIPACAO P
ON PC.COD_PROFISSIONAL_CINEMA = P.COD_PROFISSIONAL_CINEMA
WHERE PC.NOME = 'Bruce Willis';

SELECT F.NOME
FROM PROFISSIONAL_CINEMA PC INNER JOIN PARTICIPACAO P
ON PC.COD_PROFISSIONAL_CINEMA = P.COD_PROFISSIONAL_CINEMA
INNER JOIN FILME F ON F.COD_FILME = P.COD_FILME
WHERE PC.NOME = 'Bruce Willis';
```

Saída do Script x Resultado da Consulta x

Todas as Linhas Extraídas: 2 em 0,029 segundos

NOME	
1	Corpo Fechado
2	Sexto Sentido

id
1
2
3

UNION

id
2
3
4



id
1
2
3
4

Append
result sets
vertically

id
1
2
3

INNER
JOIN

id
2
3
4



id	id
2	2
3	3

Append
result sets
horizontally

Junção vs Union



EXEMPLO PRÁTICO - MySQL

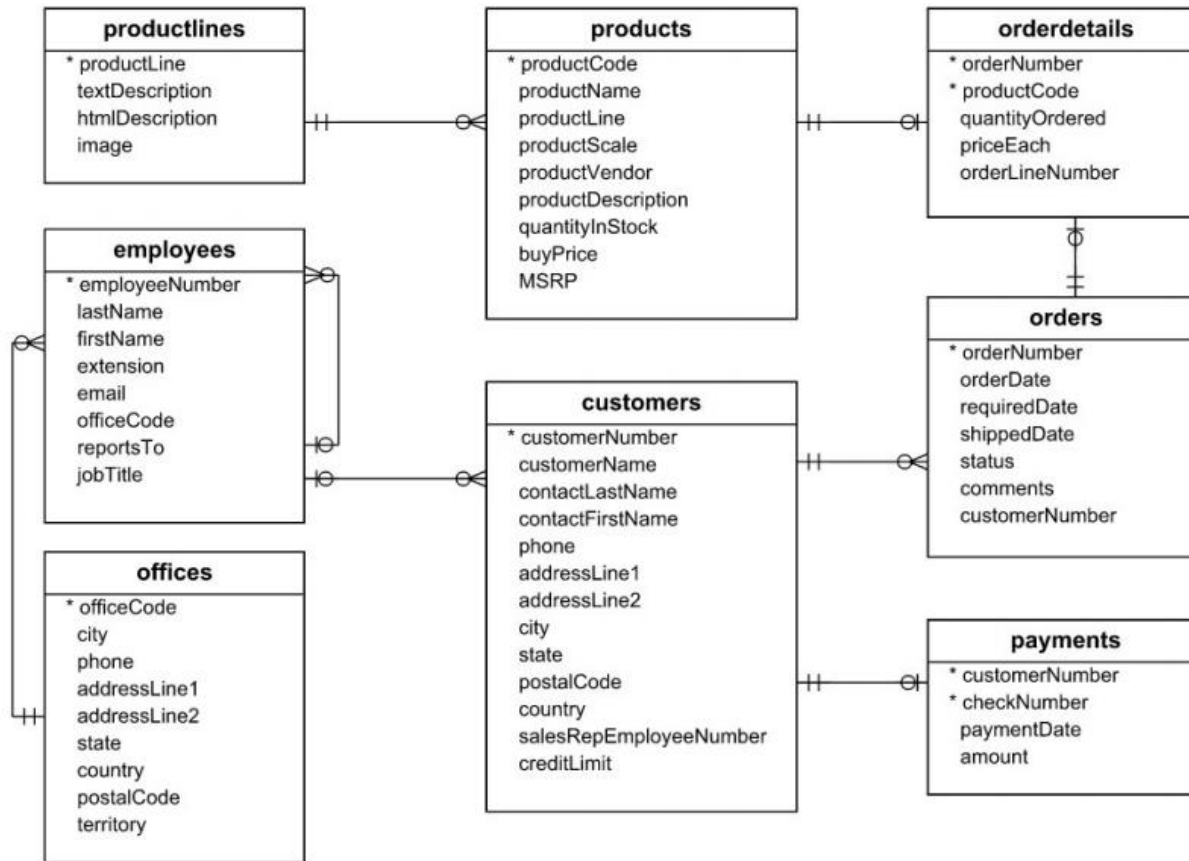
Orientação

Acesso o phpMyAdmin disponível em: <https://vmi578219.contaboserver.net/phpMyAdmin>

- Usuário e senha fornecido pelo professor via chat do Zoom.
- A base de dados a realizar consultas chama-se: classicmodels

Testar o acesso online.

MySQL Sample Database Diagram



Esquema de
Dados de
Exemplo

O esquema da base de dados de exemplo contém as seguintes tabelas:

Customers: Armazena dados de clientes

Products: Armazena uma lista de modelos de carros

ProductLines: Armazena uma lista de linhas de produtos e suas categorias

Orders: Armazena dados de transações de clientes

OrderDetails: Armazena os itens envolvidos em uma transação de compra

Payments: Armazena os pagamentos feitos pelos clientes

Employees: Armazena dados dos empregados e também dados organizacionais, tais como hierarquia

Offices: Armazena dados de venda por escritório

Esquema de Dados de Exemplo

Consultando a tabela 'employees'

Listar todos os empregados pelo sobrenome:

- `SELECT lastName FROM employees;`

Listar todos os empregados contendo sobrenome, nome e cargo:

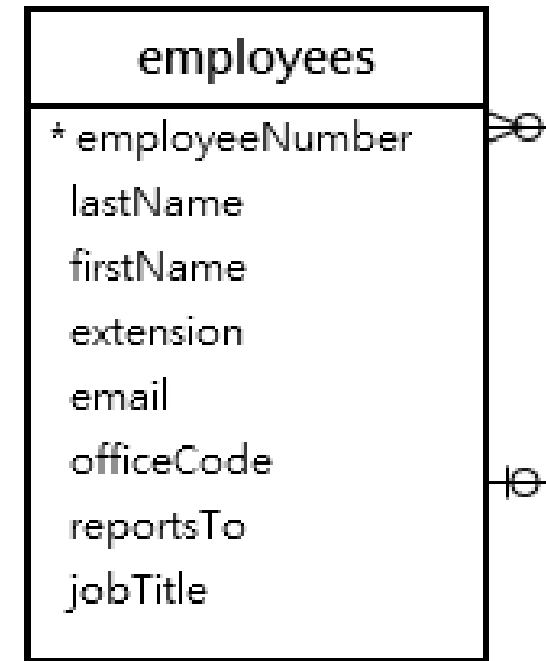
- `SELECT lastName, firstName, jobTitle FROM employees;`

Listar todos os empregados contendo matrícula, sobrenome, nome, extensão, código do escritório, chefe imediato e cargo

- `SELECT employeeNumber, lastName, firstName, extension, email, officeCode, reportsTo, jobTitle FROM employees;`

Listar todos os empregados com todos os atributos disponíveis:

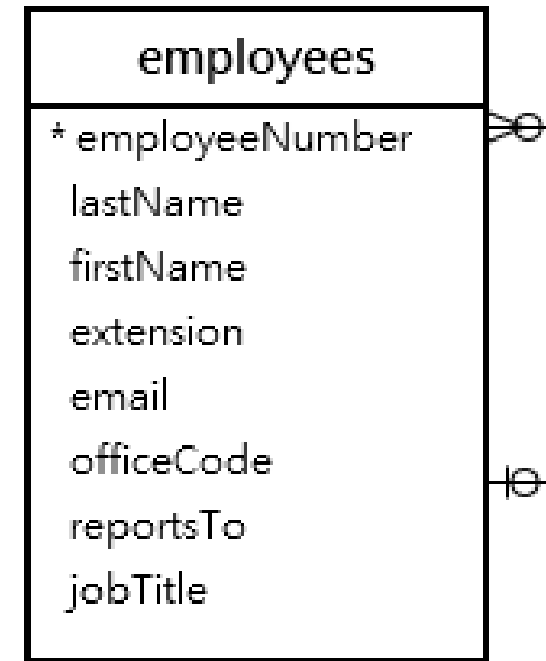
- `SELECT * FROM employees;`



Consultando a tabela 'employees'

Listar todos os empregados e seus atributos que contenham primeiro nome Jeff e também Diane.

- `SELECT * FROM employees WHERE firstName = "Jeff" UNION`
`SELECT * FROM employees WHERE firstName = "Diane"`



Consultando a tabela 'customers'

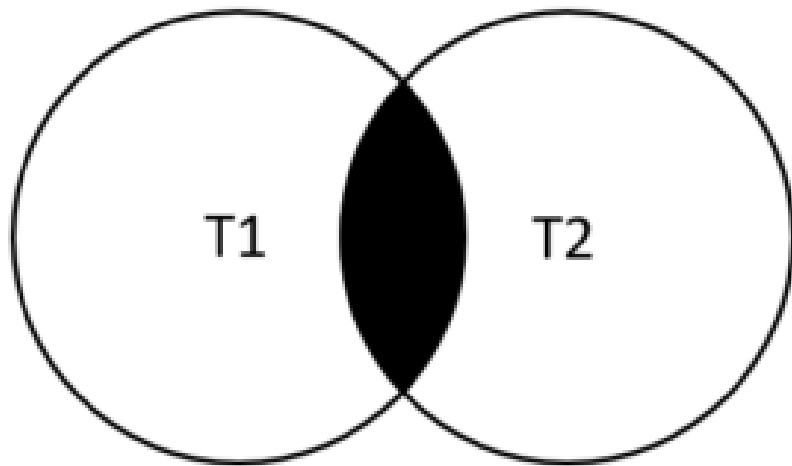
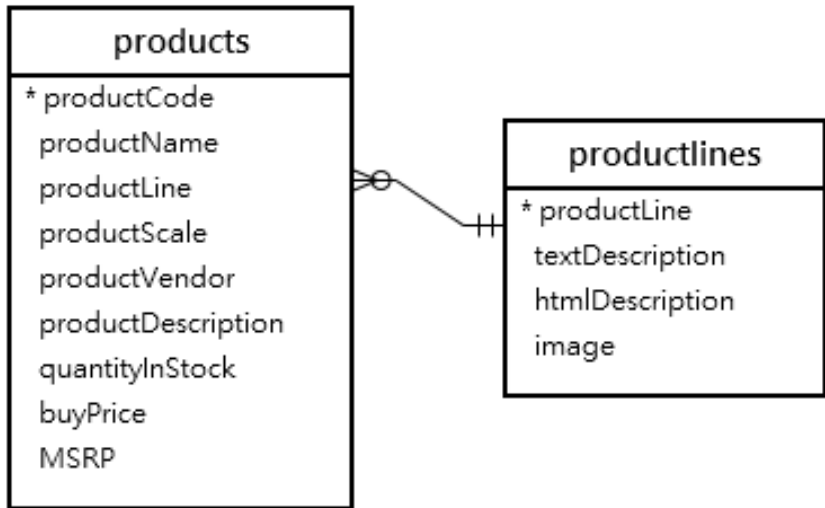
```
SELECT customername, country FROM customers WHERE  
country = 'USA' OR country = 'France';
```

```
SELECT customername, country, creditLimit FROM customers  
WHERE(country = 'USA' OR country = 'France') AND creditlimit >  
100000;
```

```
SELECT customername, country, state FROM customers WHERE  
country = 'USA' AND state = 'CA';
```

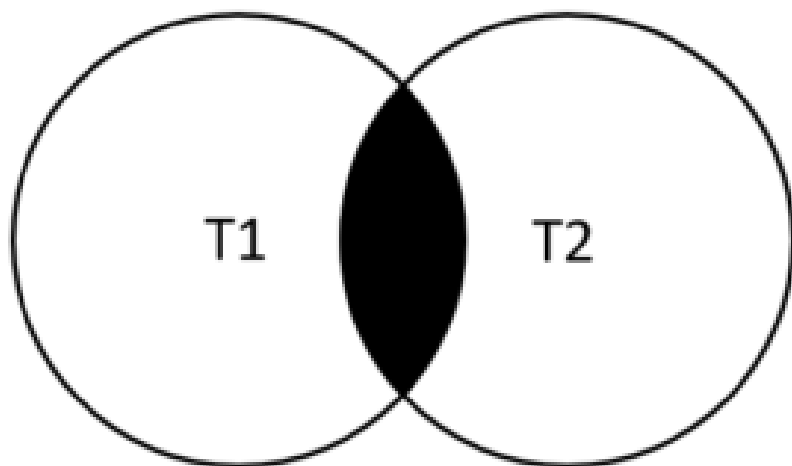
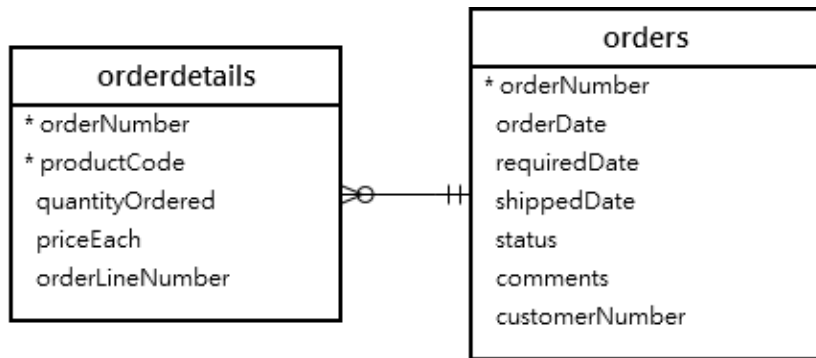
```
SELECT customername, country, state, creditlimit FROM  
customers WHERE country = 'USA' AND state = 'CA' AND  
creditlimit > 100000;
```

customers
* customerNumber
customerName
contactLastName
contactFirstName
phone
addressLine1
addressLine2
city
state
postalCode
country
salesRepEmployeeNumber
creditLimit



Consultando as tabelas 'products' e 'productlines'

```
SELECT productCode, productName, textDescription FROM  
products t1 INNER JOIN productlines t2 ON t1.productline =  
t2.productline;
```



Consultando as tabelas 'orderdetails' e 'orders'

```
SELECT
    t1.orderNumber,
    t1.status,
    SUM(quantityOrdered * priceEach) total

FROM
    orders t1

INNER JOIN
    orderdetails t2
        ON t1.orderNumber = t2.orderNumber

GROUP BY orderNumber;
```

Exercícios

Acesso o phpMyAdmin disponível em: <https://vmi578219.contaboserver.net/phpMyAdmin>

- Usuário e senha fornecido pelo professor via chat do Zoom.
- A base de dados a realizar consultas chama-se: classicmodels

Executar as consultas abaixo:

- Listar o endereço, cidade, estado e país de todos os “offices” (escritório)
- Listar o primeiro nome, último nome e também o cargo dos empregados
- Listar o primeiro nome, ultimo nome e cargo de empregados mas somente que trabalhem em NYC

Exercícios

Executar as consultas abaixo:

- Listar todos os produtos cujos preços estão entre 90 e 100
 - DICA: Pode usar função BETWEEN ou operadores lógicos maior que e menor que
- Listar todos os produtos cujos os preços NÃO estão entre 20 e 100
- Listar o nome e país de todos os clientes que possuem um consultor de venda (coluna: salesrepemployeenumber)
 - DICA: Precisa usar a função NULL
 - Consegue mostrar o número do consultor de venda em uma coluna?

Exercícios

Executar as consultas abaixo:

- Listar o primeiro nome e sobrenome de empregados e clientes na mesma tabela.
 - DICA: Precisa usar a função UNION

DESAFIOS:

- Listar o nome completo de clientes e empregados em uma única coluna.
 - DICA: Precisa usar as funções UNION e CONCAT

Alexander Semenov
Allen Nelson

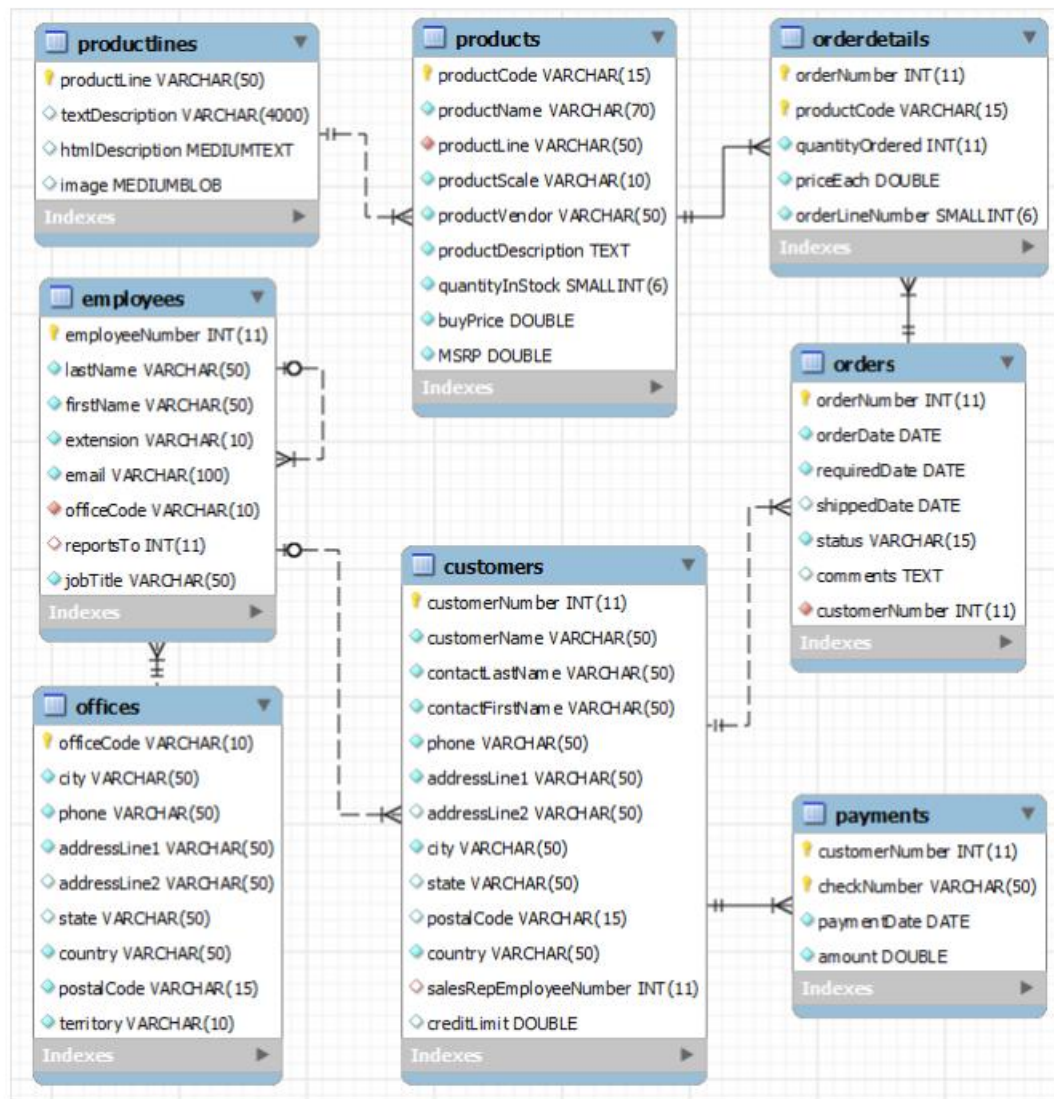
- Como podemos repetir a consulta acima porém de forma ordenada?
 - DICA: Precisa usar a função ORDER BY

- Como separar os clientes e empregados com uma coluna de identificação?

- EXEMPLO:

Adrian Huxley	Customer
Akiko Shimamura	Customer
Alejandra Camino	Customer

Esquema para consulta



- 🔑 Key: (Part of) Primary Key
- ◆ Filled Diamond: NOT NULL
- ◇ Not filled Diamond: NULL
- 🔴 Red colored: (Part of) Foreign key
- 🔵 Blue lined Diamond: Simple attribute (no key)

Can be combined for example:

- 🔴 is a Red colored Key so it's a Primary Key which is also a Foreign Key
- 🔑 is a Yellow (non Red) Key so it's only a Primary Key
- ◆ is a blue lined filled diamond so it's a NOT NULL simple attribute
- 🔴 is a red colored filled diamond so it's a NOT NULL Foreign Key
- ◇ is a blue lined not filled diamond so it's a simple attribute which can be NULL
- 🔴 is a red colored not filled diamond so it's a Foreign Key which can be NULL

A close-up photograph of a man with dark, curly hair and a prominent mustache. He is smiling broadly, showing his teeth, and giving two thumbs up with both hands. He is wearing a light blue shirt and a grey jacket. The background is blurred with bokeh lights. The image has a dark overlay with white text.

Na próxima aula...

Falaremos um pouco mais sobre Big Data Analytics.