SQL – PARTE 2

Sérgio Mergen

Esquema de Exemplo

Projeto (idProj, nome, duracao, custo, idDepto)

idDepto referencia depto

Funcionario (*idFunc, nome, salario, idDepto, idChefe*)

idDepto referencia depto

idChefe referencia Funcionario

Depto (idDepto, nome, predio, idDiretor)

idDiretor referencia Funcionario

Alocacao (idProj, idFunc, funcao)

idProj referencia projeto

idFunc referencia funcionario

DQL - Data Query Language

- Permite especificar consultas para acessar os registros armazenados nas tabelas.
- As consultas são compostas por:
  - Seleções de atributos de tabelas.
  - Filtros sobre valores de atributos.
  - Junções entre tabelas
  - Agrupamentos de atributos
  - ...

Estrutura básica de uma consulta

- Um SQL típico tem a forma:

select A_1, A_2, ..., A_n
from r_1, r_2, ..., r_m
where P

  - A_i representa um atributo
  - R_i representa uma relação
  - P é um predicado.

- O resultado da consulta é uma relação.

Cláusula Select

- A cláusula **select** lista os atributos desejados como resultado da consulta

- Um asterisco na cláusula **select** denota “todos atributos”

- Ex.

```
select *
from projeto
```

- NOTA: SQL é case insensitive para a maioria dos bancos
  - E.g. *Projeto* ≡ *projeto* ≡ *PROJETO*

Cláusula Select

- Exemplo: encontre os nomes e a duração de todos projetos:

Projeto				
idProj	**nome**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	1

select *nome, duracao*
from *projeto*

Resposta	
nome	**duracao**
ABC	3
Lucrei	2

Cláusula Select

- SQL permite dados duplicados como resposta de consultas.
- Para forçar a eliminação de duplicatas, use a palavra chave **distinct** depois do **select**.
- A palavra chave **all** especifica que duplicatas não serão removidas.

Cláusula Select

- Ex. Encontre os nomes de todos os prédios, sem repetir os nomes

Depto		
idDepto	**nome**	**predio**
1	TI	Centro – 3
2	Marketing	Norte - 2
3	RH	Centro – 3

select distinct *predio*
from *depto*

Resposta
predio
Centro – 3
Norte - 2

Cláusula Select

- A clausula **select** pode conter expressões aritméticas envolvendo os operadores +, –, ∗, e /.

- Funções também são permitidas na cláusula **select**, como as funções de manipulação de string

Cláusula Select

- Retornar o nome dos funcionários e o seu salário anual

Funcionario			
idFunc	**nome**	**salario**	**idDepto**
1	Marcos	3.000	1
2	Ana	4.000	2
3	João	2.500	1

select *nome, salario* ∗ 12
from *funcionario*

Resposta	
nome	**salario * 12**
Marcos	36.000
Ana	48.000
João	30.000

Cláusula Where

- A clausula **where** especifica condições que a consulta deve satisfazer
- As condições podem usar as seguintes operações:
  - >
  - >=
  - <=
  - <>
  - Is null
  - Is not null
- Comparações podem ser combinadas usando os conectores **and**, **or**, e **not**.
- Comparações podem ser aplicadas ao resultado de expressões aritméticas.

Cláusula Where

- Ex. Encontre todos projetos com duração estimada de 3 anos e cujo custo estimado esteja acima de R$12.000.

Projeto				
idProj	**nome**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	1
3	Caos	3	50.000	3

select *
from *projeto*
where *duracao* = *3* **and** *custo* > 12000

Resposta				
idProj	**nome**	**duracao**	**custo**	**idDepto**
3	Caos	3	50.000	3

Cláusula Where

- SQL possui o operador de comparação **between**
- Usado para substituir comparações usando os operadores convencionais
- Ex. os dois abaixo são iguais
  - *C1* **between** *v1* **and** *v2*
  - *c1* **>=** *v1* **and** *c1* **<=** *v2*

Cláusula Where

- Encontre o nome dos projetos cujo custo estimado esteja entre $30.000 e $50.000 (isto é, ≥ $30,000 e ≤ $50,000)

Projeto				
idProj	**nome**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	1
3	Caos	3	50.000	3

```
select *
from projeto
where custo between 30000 and 50000
```

Resposta				
idProj	**nome**	**duracao**	**custo**	**idDepto**
2	Lucrei	2	30.000	1
3	Caos	3	50.000	3

Cláusula From

- A clausula **from** especifica as relações envolvidas na consulta

- Caso mais de uma relação estiver presente
  - a clausula **where** pode ser usada para encontrar correspondências entre as tuplas

Exemplo

- Encontre o produto cartesiano dos projetos e departamentos

Projeto				
idProj	**nome**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	1

Depto		
idDepto	**nome**	**predio**
1	TI	3
2	Marketing	2

Exemplo

- Encontre o produto cartesiano dos projetos e departamentos

Projeto				
idProj	**nome**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	1

Depto		
idDepto	**nome**	**predio**
1	TI	3
2	Marketing	2

select *

from *depto, projeto*

Resposta							
idProj	**nome**	**duracao**	**custo**	**idDepto**	**idDepto**	**nome**	**predio**
1	ABC	3	12.000	1	1	TI	3
1	ABC	3	12.000	1	2	Marketing	2
2	Lucrei	2	30.000	1	1	TI	3
2	Lucrei	2	30.000	1	2	Marketing	2

Exemplo

- Encontre departamentos e seus projetos

Projeto				
idProj	**nome**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	1

Depto		
idDepto	**nome**	**predio**
1	TI	3
2	Marketing	2

select *

from *depto, projeto*

where *depto.idDepto = projeto.iddepto*

Resposta							
idProj	**nome**	**duracao**	**custo**	**idDepto**	**idDepto**	**nome**	**predio**
1	ABC	3	12.000	1	1	TI	3
2	Lucrei	2	30.000	1	1	TI	3

Exemplo

- Encontre departamentos e seus projetos

Projeto				
idProj	**nome**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	1

Depto		
idDepto	**nome**	**predio**
1	TI	3
2	Marketing	2

select *projeto.nome, depto.nome*

from *depto, projeto*

where *depto.idDepto = projeto.iddepto*

Resposta	
Projeto.nome	**Depto.nome**
ABC	TI
Lucrei	TI

Cláusula From

- Encontre o nome de cada departamento e o nome de todos os respectivos projetos desses departamentos **com duração estimada de 3 anos.**

Projeto				
idProj	**nome**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	1

Depto		
idDepto	**nome**	**predio**
1	TI	3
2	Marketing	2

Cláusula From

- Encontre o nome de cada departamento e o nome de todos os respectivos projetos desses departamentos **com duração estimada de 3 anos.**

Projeto				
idProj	**nome**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	1

Depto		
idDepto	**nome**	**predio**
1	TI	3
2	Marketing	2

select *depto.nome, projeto.nome*

from *depto, projeto*

where *depto.idDepto = projeto.iddepto*

and *duracao = 3*

Resposta	
Depto.nome	**Projeto.nome**
TI	ABC

A operação de Renomeação

- A linguagem SQL permite a renomeação de relações e atributos recorrendo à cláusula **as** :

  old_name **as** *new_name*

- Caso se pretenda utilizar um nome com espaços, esse nome deverá ser colocado entre **aspas**.

A operação de Renomeação

- Encontre o nome de cada departamento e de seus respectivos diretores, renomeando as colunas para DEPTO e DIRETOR

Funcionario			
idFunc	**nome**	**salario**	**idDepto**
1	Marcos	3.000	1
2	Ana	4.000	2
3	João	2.500	1

Depto			
idDepto	**nome**	**predio**	**idDiretor**
1	TI	3 - centro	1
2	RH	2 - norte	2

A operação de Renomeação

- Encontre o nome de cada departamento e de seus respectivos diretores, renomeando as colunas para DEPTO e DIRETOR

Funcionario			
idFunc	**nome**	**salario**	**idDepto**
1	Marcos	3.000	1
2	Ana	4.000	2
3	João	2.500	1

Depto			
idDepto	**nome**	**predio**	**idDiretor**
1	TI	3 - centro	1
2	RH	2 - norte	2

select *depto.nome as DEPTO, funcionario.nome as DIRETOR*
from *depto, funcionario*
where *depto.idDiretor = funcionario.idFunc*

Resposta	
DEPTO	**DIRETOR**
TI	Marcos
RH	Ana

Variáveis de tupla

- Variáveis de tupla servem para diferenciar instâncias de tabelas diferentes usadas em uma mesma consulta

- As variáveis de tupla são definidas na cláusula **from** por intermédio da cláusula **as**.

- Cláusula **as** é opcional e pode ser omitida

Variáveis de tupla

- Para cada projeto, apresente seu nome, nome dos funcionários alocados e a função desses funcionários no projeto

Funcionario	
idFunc	**nome**
1	Marcos
2	Ana
3	João

Projeto	
idProj	**nome**
1	Lucrei
2	Caos

Alocação		
idProj	**idFunc**	**função**
1	1	Coordenador
1	3	Analista

Variáveis de tupla

- Para cada projeto, apresente seu nome, nome dos funcionários alocados e a função desses funcionários no projeto

Funcionario	
idFunc	**nome**
1	Marcos
2	Ana
3	João

Projeto	
idProj	**nome**
1	Lucrei
2	Caos

Alocação		
idProj	**idFunc**	**função**
1	1	Coordenador
1	3	Analista

select *p.nome, f.nome, funcao*
from *projeto* **as** *p, funcionario* **as** *f, alocacao a*
where *a.idProj = p.idProj*
and *a.idFunc = f.idFunc*

Resposta		
p.nome	**f.nome**	**Funcao**
Lucrei	Marcos	Coordenador
Lucrei	João	Analista

Variáveis de tupla (cont)

- As variáveis de tuplas podem ser usadas para criar várias cópias de uma mesma relação
- Usada quando a mesma tabela aparece mais de uma vez na cláusula from

Variáveis de tupla (cont)

- Apresente o nome do funcionário e de seu respectivo chefe

Funcionario		
idFunc	**nome**	**idChefe**
1	Marcos	null
3	João	1

Variáveis de tupla (cont)

- Apresente o nome do funcionário e de seu respectivo chefe

Funcionario		
idFunc	**nome**	**idChefe**
1	Marcos	null
3	João	1

A tabela Funcionario precisa ser instanciada duas vezes

Sub: para identificar subordinados
Chefe: para identificar chefes

Essas instâncias são as variáveis de tupla

sub

idFunc	nome	IdChefe
1	Marcos	null
3	João	1

chefe

idFunc	nome	IdChefe
1	Marcos	null
3	João	1

Variáveis de tupla (cont)

- Apresente o nome do funcionário e de seu respectivo chefe

Funcionario		
idFunc	**nome**	**idChefe**
1	Marcos	null
3	João	1

Nem todo registro na variável de tupla **sub** será efetivamente um subordinado.

Para ser subordinado, o registro deve ter algum conteúdo na coluna idChefe

No exemplo, apenas o **Joao** é **subordinado**

sub

idFunc	nome	IdChefe
1	Marcos	null
3	**João**	1

chefe

idFunc	nome	IdChefe
1	Marcos	null
3	João	1

Variáveis de tupla (cont)

- Apresente o nome do funcionário e de seu respectivo chefe

Funcionario		
idFunc	**nome**	**idChefe**
1	Marcos	null
3	João	1

Da mesma forma, nem todo registro na variável de tupla **chefe** será efetivamente um chefe.

Para ser chefe, o registro deve possuir um idFunc que é usado como idChefe de outro registro.

No exemplo, apenas o **Marcos** é **chefe**

sub

idFunc	nome	IdChefe
1	Marcos	null
3	João	1

chefe

idFunc	nome	IdChefe
1	**Marcos**	null
3	João	1

Variáveis de tupla (cont)

- Apresente o nome do funcionário e de seu respectivo chefe

Funcionario		
idFunc	**nome**	**idChefe**
1	Marcos	null
3	João	1

Para cada subordinado, deve-se encontrar seu chefe.

Isso é feito usando o seguinte critério de junção
Sub.idChefe = chefe.idFunc

sub

idFunc	nome	IdChefe
1	Marcos	null
3	João	1

chefe

idFunc	nome	IdChefe
1	Marcos	null
3	João	1

Variáveis de tupla (cont)

- Apresente o nome do funcionário e de seu respectivo chefe

Funcionario		
idFunc	**nome**	**idChefe**
1	Marcos	null
3	João	1

O produto cartesiano cruza os registros das duas variáveis de tupla

```
select  *
from    func as chefe, func as sub
```

chefe

sub

idFunc	**nome**	**idChefe**	**idFunc**	**nome**	**IdChefe**
1	Marcos	null	1	Marcos	null
1	Marcos	null	3	João	1
1	João	1	1	Marcos	null
1	João	1	3	João	null

Variáveis de tupla (cont)

- Apresente o nome do funcionário e de seu respectivo chefe

Funcionario		
idFunc	**nome**	**idChefe**
1	Marcos	null
3	João	1

O critério de junção mantem apenas as combinações que efetivamente relacionem o subordinado com o seu chefe

```
select distinct  sub.nome as empregado, chefe.nome as chefe
from             func as chefe, func as sub
where            chefe.idFunc = sub.idChefe
```

Resposta	
empregado	**chefe**
João	Marcos

Operações com Strings e LIKE

- SQL inclui um mecanismo de concordância de padrões para comparações envolvendo strings.

- Os padrões são descritos recorrendo a dois caracteres especiais:
  - percentagem(%).
    - O caracter % concorda com qualquer substring.
  - Sublinhado(_).
    - O caracter _ concorda com qualquer caractere.

- SQL suporta uma variedade de operações com strings, tais como:
  - concatenação
  - conversão de maiúsculas para minúsculas (e vice versa)
  - cálculo o comprimento, extração de substrings, etc.,

Operações com Strings e LIKE

- Ex. Listar todos os nomes de departamentos cuja descrição do prédio inclua a subcadeia “centro”.

Depto		
idDepto	**nome**	**predio**
1	TI	3 – centro
2	RH	5 - centro
3	Marketing	3 - norte

select *
from *depto*
where *predio* **like** ‘%centro%’

Resposta		
idDepto	**nome**	**predio**
1	TI	3 – centro
2	RH	5 - centro

Ordenando as tuplas

- A cláusula **order by** é usada para ordenar os registros

- Para cada atributo, pode-se especificar
  - **desc** para ordenação descente
  - **asc** para ordenação ascendente
  - por default, assume-se ordem ascendente.

- E.g. **order by** *p.nome* **desc**

- Pode-se ter mais do que uma chave de ordenação, separando-as com vírgulas

Ordenando as tuplas

- Ex. Listar em ordem de duração os nomes de todos os projetos do departamento de TI

Projeto				
idProj	**nome**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	1
3	Show	1	10.000	2

Depto		
idDepto	**nome**	**predio**
1	TI	3
2	Marketing	2

select *
from *projeto p, depto d*
where *p.idDepto = d.idDepto* **and** *d.nome* = 'TI'
order by *duracao*

Resposta							
idProj	**nome**	**duracao**	**custo**	**idDepto**	**idDepto**	**nome**	**predio**
2	Lucrei	2	30.000	1	1	TI	3
1	ABC	3	12.000	1	1	TI	3

Funções de Agregação

- Estas funções aplicam-se a conjuntos de valores de uma coluna de uma relação, devolvendo um valor

avg: valor médio
min: valor mínimo
max: valor máximo
sum: soma dos valores
count: número de valores

Funções de Agregação (cont.)

- Determinar o salário médio dos funcionários.

Funcionario		
idFunc	**nome**	**salario**
1	Marcos	10000
2	Ana	6000
3	João	5000

select avg *(salario)*
from *funcionario*

Resposta
Avg(salario)
7000

Funções de Agregação (cont.)

- Calcular o número de departamentos.

Depto		
idDepto	**nome**	**idDiretor**
1	TI	1
2	RH	5
3	Marketing	3

select count (*)
from *depto*

Resposta
Count(*)
3

Funções de Agregação (cont.)

- Encontrar o número de funcionários alocados em projetos.

Alocação		
idProj	**idFunc**	**função**
1	1	Coordenador
1	3	Analista
2	1	Coordenador

select count (distinct *idFunc)*
from *alocacao*

Resposta
Count(distinct idFunc)
2

Funções de agregação – Group By

- A cláusula **group by** realiza a agregação para subconjuntos das tuplas do resultado
- Ex.

```
select c1, c2, avg (c3)
from t1
group by c1, c2
```

Nota: Atributos na cláusula **select** fora de funções de agregação têm de aparecer na lista **group by**

Nota: Se aparecer mais do que um atributo em **group by**, então cada grupo é formado pelas tuplas com valores iguais *em todos* esses os atributos

Funções de agregação – Group By

- Ex. Listar o salário médio por departamento.

Funcionario			
idFunc	**nome**	**Salario**	**idDepto**
1	Marcos	10000	1
2	Ana	6000	2
3	João	5000	1

select *idDepto, avg (salario)*
from *funcionario*
group by *idDepto*

Resposta	
idDepto	**avg (salario)**
1	7500
2	6000

Funções de agregação – Group By

- A cláusula **having** realiza a agregação para subconjuntos das tuplas do resultado
- Ex.

```
select    c1, c2, avg (c3)
from      t1
group by  c1, c2
having    avg (c3) op valor
```

Nota: predicados na cláusula **having** são aplicados depois da formação dos grupos. Já predicados na cláusula **where** são aplicados antes da formação dos grupos.

Funções de Agregação – Cláusula Having

- Ex. Listar os nomes de todas os departamentos cuja média de salário seja superior a R$5.000.

Funcionario			
idFunc	**nome**	**salario**	**idDepto**
1	Marcos	10000	1
2	Ana	6000	2
3	João	5000	1

Depto			
idDepto	**nome**	**predio**	**idDiretor**
1	TI	3 - centro	1
2	RH	2 - norte	2

select *d.nome,* **avg** *(salario)*
from *funcionario f, depto d*
where *f.idDepto = d.idDepto*
group by *f.idDepto*
having avg *(salario)* > 7000

Resposta	
nome	**Avg(salario)**
TI	7500

Atividade Individual

- Execute o script fornecido para criar o banco de dados de filmes no MySQL.

- Resolva as consultas solicitadas no moodle.
  - Apenas as **10 primeiras consultas** serão avaliadas
  - As seis últimas não serão avaliadas, mas podem ser entregues junto com as 10 primeiras
    - É interessante tentar resolvê-las
    - Elas podem explorar comandos não vistos durante a aula
    - Essa é uma boa forma de aprender mais a respeito dos recursos da linguagem SQL

Atividade Individual

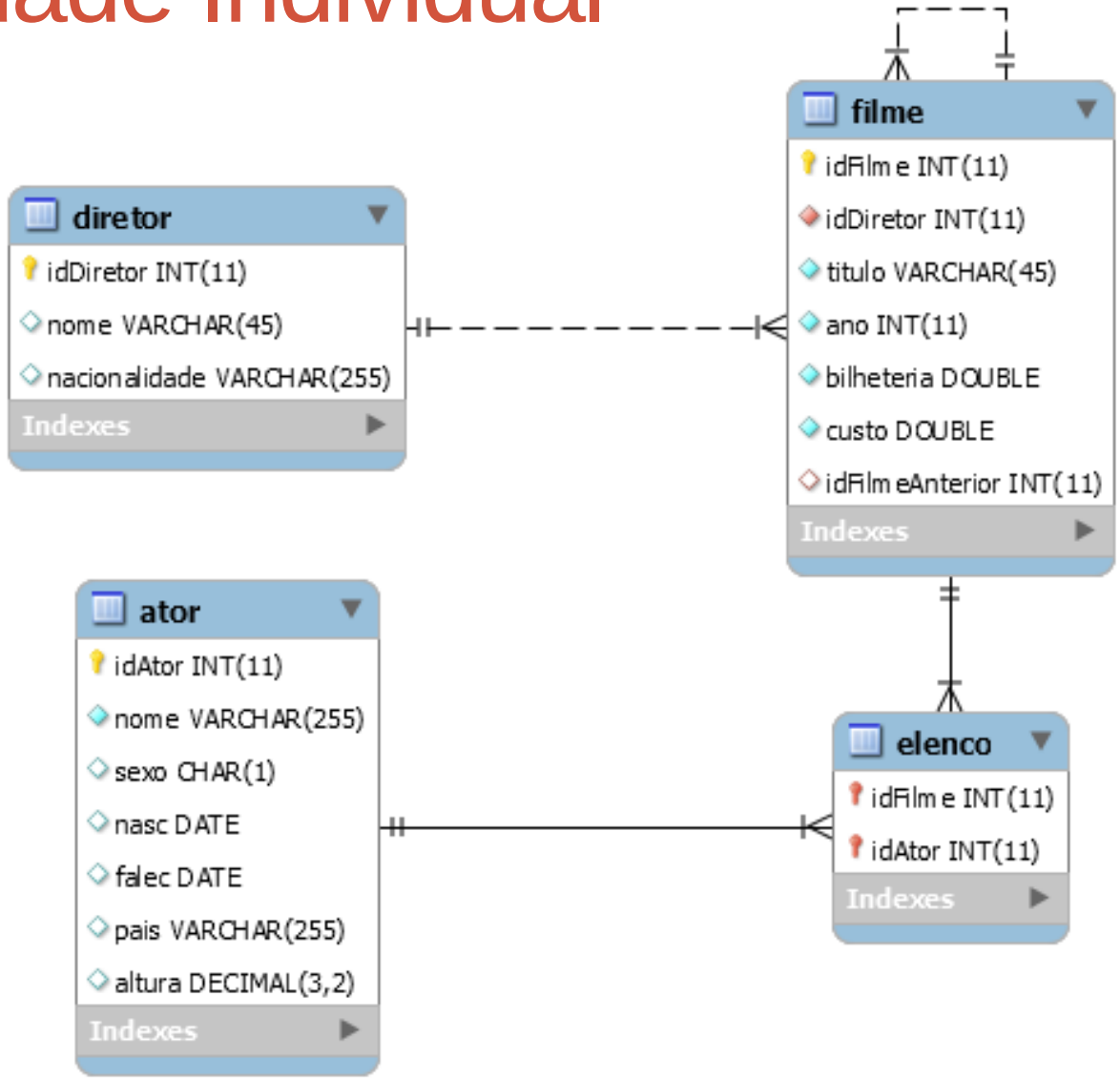
- Entregue um arquivo PDF contendo, para cada consulta:
  - O número sequencial da consulta
  - O texto do que foi solicitado
  - O código SQL

- Ex.

```
-- 1. Selecione título de filmes que comecem com 'batman'.
SELECT ...

-- 2. Exibir o título e ano de filmes lançados entre 2010 e 2015.
SELECT ...
```

Atividade Individual



SQL – PARTE 2