



Universidade Federal do Ceará
Campus Crateús

Fundamentos de Banco de Dados

Aula 10 - Consultas



Professora Vitória Regina - vitoria@crateus.ufc.br



O que estudaremos?

- Consultas.



Consultas

- As consultas são descritas na linguagem SQL através da operação **SELECT**;
- Geralmente, uma consulta tem a forma:
SELECT <lista_de_atributos>
FROM <lista_de_tabelas>
WHERE <condição>

Consultas

- Onde:
 - **<lista_de_atributos>** corresponde às colunas que devem ser selecionadas das tabelas:
 - Corresponde à operação de projeção da álgebra relacional.
 - **<lista_de_tabelas>** corresponde às tabelas que serão usadas para realizar a consulta.

Consultas

- Onde:
 - **<condição>** é uma expressão booleana que indica o critério de seleção das tuplas:
 - Pode ser omitida, quando quisermos recuperar todas as tuplas de uma tabela;
 - Os operadores relacionais =, <, ≤, >, ≥ e ≠ podem ser usados para expressar as condições;
 - Os conectores lógicos AND, OR e NOT podem ser usados para especificar condições compostas.

Consultas

- Vamos considerar a seguinte relação Empregado:

Matrícula	Nome	Salário	Supervisor	CodDepartamento
1111-1	João	2500,00	1111-4	1
1111-2	Maria	2500,00	1111-3	2
1111-3	Carlos	4500,00	NULL	2
1111-4	Joaquim	4500,00	NULL	1

Consultas

- Vamos considerar a seguinte relação Departamento:

<u>CodDepartamento</u>	Nome	Gerente
1	Financeiro	1111-4
2	Vendas	1111-3

Consultas

- Vamos considerar a seguinte relação Projeto:

<u>CodProjeto</u>	Nome	CodDepartamento
1	Venda Fácil	2
2	MaxLucro	1
3	Cliente Ouro	1

Consultas

- Vamos considerar a seguinte relação Dependente:

Empregado	NomeDep	Parentesco
1111-2	Marcos	Filho
1111-2	Luís	Filho
1111-3	Ana	Cônjuge

Consultas

- Vamos considerar a seguinte relação TrabalhaProjeto:

Empregado	CodProjeto	NumHoras
1111-1	2	12
1111-1	3	12
1111-2	1	12
1111-2	2	12
1111-4	2	12
1111-4	3	12

Consultas

- **Exemplo 1:** Selecionar a matrícula e o nome de todos os empregados que trabalham no departamento 2.

SELECT Matricula, Nome **FROM** Empregado

WHERE CodDepartamento =2;

- **Resultado:**

Matrícula	Nome
1111-2	Maria
1111-3	Carlos

Consultas

- **Exemplo 2:** Selecionar a matrícula e o nome dos empregados do departamento 2 que ganham mais de três mil reais.

SELECT Matricula, Nome **FROM** Empregado

WHERE CodDepartamento=2 **AND** Salario>3000;

- **Resultado:**

Matrícula	Nome
1111-3	Carlos

Consultas

- Podemos omitir a condição para selecionar todas as tuplas de uma tabela:
 - **Exemplo 3:** Selecionar a matrícula e o nome de todos os empregados.

SELECT Matricula, Nome **FROM** Empregado

- **Resultado:**

Matrícula	Nome
1111-1	João
1111-2	Maria
1111-3	Carlos
1111-4	Joaquim

Consultas

- Podemos também selecionar todas as colunas de uma tabela usando o valor *.
 - **Exemplo 4:** Selecionar as informações sobre os empregados do departamento dois.

SELECT * FROM Empregado **WHERE** CodDepartamento=2;

- **Resultado:**

Matrícula	Nome	Salário	Supervisor	CodDepartamento
1111-2	Maria	2500,00	1111-3	2
1111-3	Carlos	4500,00	NULL	2

Colunas

- Podemos alterar o nome de uma ou mais colunas através da cláusula AS:
 - **Exemplo 5:** Selecionar o nome de todos os empregados.
SELECT Matricula, Nome **AS** NomeEmpregado **FROM** Empregado;
 - **Resultado:**

Matrícula	NomeEmpregado
1111-1	João
1111-2	Maria
1111-3	Carlos
1111-4	Joaquim

Consultas

- Podemos aplicar cálculos aos valores de uma coluna:
 - **Exemplo 6:** Calcule como ficaria o novo salário de cada empregado caso eles recebessem um aumento de 15%.
SELECT Matricula, Nome, $1.15 * \text{Salario}$ **AS** SalarioReajustado **FROM** Empregado;

Matrícula	Nome	SalarioReajustado
1111-1	João	2875,00
1111-2	Maria	2875,00
1111-3	Carlos	5175,00
1111-4	Joaquim	5175,00

Consultas

- Podemos também renomear tabelas para fazer junção de uma tabela com ela mesma:

- **Exemplo 07:** Recupere a matrícula, o nome e o nome do supervisor de cada empregado.

```
SELECT E.Matricula, E.Nome, S.Nome AS Supervisor  
FROM Empregado E, Empregado S  
WHERE E.Supervisor=S.Matricula
```

Consultas

- **Exemplo 07:**
 - **Resultado:**

Matrícula	Nome	Supervisor
1111-1	João	Joaquim
1111-2	Maria	Carlos

Consultas

- Comparando com valores nulos:
 - Uma tabela pode conter valores nulos para vários atributos;
 - Para selecionar tuplas que possuem um valor de atributo nulo ou não nulo usamos os operadores IS e IS NOT, respectivamente;
 - **Exemplo 08:** Recupere o nome de todos os empregados que não tem supervisor.

```
SELECT Nome FROM Empregado  
WHERE Supervisor IS NULL;
```

Consultas

- Comparando com valores nulos:
 - **Exemplo 08:**
 - **Resultado:**

Nome
Carlos
Joaquim

Consultas

- Representando tabelas como conjuntos:
 - A linguagem SQL trata uma tabela como um multiconjunto:
 - Tuplas repetidas não são descartadas do resultado final.
 - Podemos usar a palavra-chave **DISTINCT** para eliminar tuplas repetidas do resultado;
 - Podemos usar a palavra-chave **ALL** para especificar explicitamente que todas as tuplas devem ser recuperadas no resultado de uma consulta.

Consultas

- Representando tabelas como conjuntos:
 - **Exemplo 09:** Recupere o salário de cada funcionário da empresa.
 - **SELECT** Salario **FROM** Empregado;
 - **Resultado:**

Salário
2500,00
2500,00
4500,00
4500,00

Consultas

- Representando tabelas como conjuntos:
 - **Exemplo 10:** Selecione todos os salários distintos pago pela empresa.
SELECT DISTINCT Salario **FROM** Empregado;
 - **Resultado:**

Salario
2500,00
4500,00

Consultas

- Representando tabelas como conjuntos:
 - SQL define três operadores para trabalhar com conjuntos:
 - **UNION** (operação de união);
 - **INTERSECT** (operação de interseção);
 - **EXCEPT** (operação de diferença).
 - As tabelas devem ser compatíveis de união para realizar qualquer uma destas operações;
 - Estas operações descartam as tuplas repetidas do resultado.

Consultas

- Representando tabelas como conjuntos:
 - **Exemplo 11:** Selecione a matrícula de todos os empregados que trabalham no departamento 1 ou que trabalham em um projeto:
(**SELECT** Matricula **FROM** Empregado **WHERE** CodDepartamento=1)
UNION
(**SELECT DISTINCT** Empregado **AS** Matricula **FROM** TrabalhaProjeto)

Consultas

- Representando tabelas como conjuntos:
 - **Exemplo 11:**
 - **Resultado:**

Matrícula
1111-1
1111-2
1111-4

Consultas

- Procurando por substrings:
 - Podemos procurar por um substring em uma cadeia de caracteres usando o operador **LIKE**.
 - Podemos usar o caractere % para substituir zero ou mais caracteres;
 - Como o caractere * usado para definir expressões regulares;
 - Exemplo: Para localizar strings que contenham o substring 'Cajazeiras' usamos a expressão '%Cajazeiras%'

Consultas

- Procurando por substrings:
 - Podemos usar o caractere _ para substituir um único caractere;
 - **Exemplo:** Para localizar todos os strings cujo segundo caractere seja a letra 'a', usamos a expressão '_a%'.
 - **Exemplo 12:** Recupere a matrícula e o nome de todos os empregados que possuem o nome Maria.

```
SELECT Matricula, Nome FROM Empregado  
WHERE Nome LIKE '%Maria%';
```

Consultas

- Procurando por substrings:
 - **Exemplo 12:**
 - **Resposta:**

Matrícula	Nome
1111-2	Maria

Consultas

- Ordenando o resultado de uma consulta:
 - Podemos ordenar o resultado de uma consulta usando o operador **ORDER BY**;
 - Podemos especificar uma lista de atributos para a ordenação, onde o primeiro é usado como primeiro critério e os demais sucessivamente, para ordenar tuplas que apresentam o mesmo valor para o critério anterior;
 - Para cada critério podemos usar os valores **ASC** ou **DESC**, para especificar se ordenação deve ser ascendente ou descendente;
 - A ordenação default é ascendente.

Consultas

- Ordenando o resultado de uma consulta:
 - **Exemplo 13:** Recupere o nome de cada empregado, juntamente com o nome do seu departamento e de cada projeto em que ele trabalha.

```
SELECT E.Nome AS Empregado, D.Nome AS Depto, P.Nome AS Projeto  
FROM Empregado E, Departamento D, Projeto P, TrabalhaProjeto TP  
WHERE E.CodDepartamento=D.CodDepartamento AND  
E.Matricula=TP.Empregado AND P.CodProjeto=TP.CodProjeto  
ORDER BY E.Nome ASC, P.Nome ASC
```

Consultas

- Ordenando o resultado de uma consulta:

- **Exemplo 13:**

- **Resultado:**

Empregado	Depto	Projeto
João	Financeiro	Cliente Ouro
João	Financeiro	Max Lucro
Joaquim	Financeiro	Cliente Ouro
Joaquim	Financeiro	Max Lucro
Maria	Vendas	Max Lucro
Maria	Vendas	Venda Facil

Consultas

- Consultas aninhadas:
 - São consultas feitas sobre os dados recuperados por uma outra consulta;
 - Usamos o operador IN para verificar se uma tupla está presente em um conjunto ou multiconjunto;
 - **Exemplo 14:** Selecione a matrícula e o nome dos empregados que trabalham no departamento gerenciado por Carlos.

Consultas

- Consultas aninhadas:

- **Exemplo 14** (continuação):

SELECT Matricula, Nome **FROM** Empregado

WHERE CodDepartamento IN

(**SELECT** E.CodDepartamento **FROM** Empregado E, Departamento D

WHERE E.Matricula=D.Gerente **AND** E.Nome='Carlos'

)

Consultas

- Consultas aninhadas:
 - **Exemplo 14** (Resultado):

Matrícula	Nome
1111-2	Maria
1111-3	Carlos

Consultas

- Consultas aninhadas:
 - Podemos usar o quantificador ALL para comparar uma tupla com um conjunto de tuplas recuperado em uma consulta aninhada;
 - O resultado da operação é verdadeiro se a comparação for verdadeira para todas as tuplas presentes no resultado da consulta aninhada;
 - **Exemplo 15:** Recupere o nome dos funcionários que ganham mais que João e Maria.

Consultas

- Consultas aninhadas:
 - **Exemplo 15** (Continuação):
SELECT Nome **FROM** Empregado
WHERE Salario > **ALL**
(**SELECT** Salario **FROM** Empregado **WHERE** Nome='João' OR
Nome='Maria')

Consultas

- Consultas aninhadas:
 - **Exemplo 15** (Continuação):
 - Obteremos o seguinte resultado:

Nome
Carlos
Joaqui m

Consultas

- Consultas aninhadas:
 - O quantificador **SOME** (ou ANY) também pode ser utilizado para comparar uma tupla com as tuplas do resultado de uma consulta aninhada;
 - O resultado da operação é verdadeiro se a comparação for verdadeira para alguma das tuplas presentes no resultado da consulta aninhada;
 - **Exemplo 16:** Recupere o nome e o salário de todos os funcionários que não trabalham no departamento 2 mas ganham mais do que algum funcionário que trabalha no departamento 2.

Consultas

- Consultas aninhadas:
 - **Exemplo 16** (continuação):
SELECT Nome **FROM** Empregado
WHERE CodDepartamento <> 2 **AND** Salario > SOME
(**SELECT** Salario **FROM** Empregado **WHERE** CodDepartamento=2)
 - **Resultado:**

Consultas

- A cláusula **EXISTS**:
 - A cláusula **EXISTS** é usada para verificar se o resultado de uma consulta aninhada é ou não vazio;
 - Ela retorna true caso o resultado da consulta não seja vazio, e false, caso contrário;
 - Podemos definir também a cláusula **NOT EXISTS**, que funciona da forma inversa;
 - **Exemplo 17:** Recupere o nome de todos os empregados que possuem algum dependente.

Consultas

- A cláusula **EXISTS**:

- **Exemplo 17** (continuação):

SELECT Nome **FROM** Empregado E

WHERE EXISTS

(**SELECT** * **FROM** Dependente D **WHERE** E.Matricula = D.Empregado)

Consultas

- A cláusula **EXISTS**:
 - **Exemplo 17** (continuação):
 - Obteremos o seguinte resultado;

Nome
Maria
Carlos

Consultas

■ Conjuntos Explícitos:

- Podemos especificar também conjuntos explícitos para realizar consultas;
- Utilizamos a cláusula IN para fazer comparações com este conjunto;
- **Exemplo 18:** Recupere o nome de todos os empregados que trabalham no projeto 1 ou no projeto 2.

Consultas

■ Conjuntos explícitos:

- **Exemplo 18** (continuação):

SELECT Nome **FROM** Empregado E, TrabalhaProjeto TR

WHERE E.Matricula = TR.Empregado **AND** TR.CodProjeto IN(1, 3);

- **Resultado:**

Nome
João
Maria
Joaquim

■ **Junção de Tabelas:**

- Até agora, vimos como fazer junção em SQL da forma clássica;
- Porém a SQL define o operador JOIN e vários tipos de junção de tabelas:
 - Equijunção;
 - Junção à esquerda;
 - Etc.

■ Junção de Tabelas – Equijunção:

- Representa a equijunção da álgebra relacional:
 - Os dois atributos de junção são considerados no resultado final
- Realizada através da cláusula **JOIN**;
- A operação de junção é definida na cláusula **FROM**;
- Sintaxe:

Tabela1 JOIN Tabela2 ON CondiçãoDeJunção

Consultas

■ Junção de Tabelas – Equijunção:

- **Exemplo 19:** Recupere o nome de todos os empregados que trabalham no departamento Financeiro.

SELECT E.Nome

FROM (Empregado E JOIN Departamento D **ON**
E.CodDepartamento=D.CodDepartamento)

WHERE D.Nome='Financeiro'

Consultas

- **Junção de Tabelas – Equijunção:**
 - **Exemplo 19** (continuação):
 - O seguinte resultado é obtido:

Nome
João
Joaquim

■ **Junção de Tabelas – Junção Externa:**

- A equijunção é uma operação de junção interna:
 - Tuplas que não estão relacionadas, ou que possuem um valor nulo para o atributo da condição de junção são descartadas do resultado final.
- Podemos incluir todos as tuplas de uma tabela no resultado de uma junção através de uma junção externa;
- A junção externa é realizada através da cláusula OUTER JOIN.

■ **Junção de Tabelas – Junção Externa:**

- Existem três tipos de junção externa:
 - Junção externa à esquerda (LEFT OUTER JOIN ou LEFT JOIN);
 - Junção externa à direita (RIGHT OUTER JOIN ou RIGHT JOIN);
 - Junção externa completa (FULL OUTER JOIN ou FULL JOIN).
- Caso os atributos de junção tenham o mesmo nome, podemos incluir a palavra-chave NATURAL antes da cláusula de junção.

■ **Junção de Tabelas – Junção Externa à Esquerda:**

- Indica que todas as tuplas da tabela do lado esquerdo da junção devem ser recuperadas:
 - Mesmo que não estejam associadas à nenhuma tupla da tabela do lado direito;
 - Mesmo que tenham um valor nulo para ao tributo da condição de junção.
- Tuplas sem nenhuma tupla correspondente à direita ou com valores nulos para o atributos de junção recebem valores nulos para todos os atributos da tabela do lado direito da junção.

Consultas

- **Junção de Tabelas – Junção Externa à Esquerda:**
 - **Exemplo 20:** Vamos supor a seguinte consulta:
 - **SELECT** *
 - **FROM** (Empregado **LEFT OUTER JOIN** Departamento **ON** Matricula=Gerente)

Consultas

■ Junção de Tabelas – Junção Externa à Esquerda:

- Resultado:

Matricula	Nome	Salario	Supervisor	CodDepo	CodDepo	Nome	Gerente
1111-1	João	2500	1111-4	1	NULL	NULL	NULL
1111-2	Maria	2500	1111-3	2	NULL	NULL	NULL
1111-3	Carlos	4500	NULL	2	2	Vendas	1111-3
1111-4	Joaquim	4500	NULL	1	1	Financeiro	1111-4

■ **Junção de Tabelas – Junção Externa à Direita:**

- Indica que todas as tuplas da tabela do lado direito da junção devem ser recuperadas:
 - Mesmo que não estejam associadas à nenhuma tupla da tabela do lado esquerdo;
 - Mesmo que tenham um valor nulo para ao tributo da condição de junção.
- Tuplas sem nenhuma tupla correspondente à esquerda ou com valores nulos para o atributos de junção recebem valores nulos para todos os atributos da tabela do lado esquerdo da junção.

Consultas

■ Junção de Tabelas – Junção Externa à Direita:

- **Exemplo 21:** Vamos supor a seguinte consulta:

SELECT *

FROM (Dependente **RIGHT OUTER JOIN** Empregado **ON**
Empregado=Matricula)

Consultas

■ Junção de Tabelas – Junção Externa à Direita

- Resultado:

Empregado	Nome Dep	Paren-tesco	Matricula	Nome	Salario	Supervisor	Depto
NULL	NULL	NULL	1111-1	João	2500	1111-4	1
1111-2	Marcos	Filho	1111-2	Maria	2500	1111-3	2
1111-2	Luís	Filho	1111-2	Maria	2500	1111-3	2
1111-3	Ana	Cônjuge	1111-3	Carlos	4500	NULL	2
NULL	NULL	NULL	1111-4	Joaquim	4500	NULL	1

■ **Junção de Tabelas – Junção Externa Completa**

- Recupera todas as tuplas das duas tabelas;
- Tuplas do lado esquerdo que não estão associadas a nenhuma tupla da tabela à direita ou que possuem valor nulo para o atributo de junção são tratadas como na junção externa à esquerda;
- Tuplas do lado direito que não estão associadas a nenhuma tupla da tabela à esquerda ou que possuem valor nulo para o atributo de junção são tratadas como na junção externa à direita.

■ Junção de Tabelas – Produto Cruzado

- Representa a operação produto cartesiano da álgebra relacional;
- Todas as tuplas da primeira tabela são combinadas com todas as tuplas da segunda tabela, independente de qualquer condição;
- Representada pela cláusula **CROSS JOIN**.

Consultas

■ Junção de Tabelas – Produto Cruzado

- **Exemplo 22:** Vamos supor a consulta abaixo:

SELECT *

FROM (Departamento **CROSS JOIN** Projeto)

Consultas

■ Junção de Tabelas – Produto Cruzado

- **Exemplo 22:** Vamos obter o seguinte resultado:

CodDepartamento	Nome	Gerente	CodProjeto	Nome	CodDepartamento
1	Financeiro	1111-4	1	Venda Fácil	2
1	Financeiro	1111-4	2	Max Lucro	2
1	Financeiro	1111-4	3	Cliente Ouro	1
2	Vendas	1111-3	1	Venda Fácil	2
2	Vendas	1111-3	2	Max Lucro	2
2	Vendas	1111-3	3	Cliente Ouro	1

Consultas

■ Funções Agregadas:

- A linguagem SQL possui algumas funções agregadas que podem ser aplicadas aos dados no momento de uma consulta:
- As principais funções são:

■ COUNT:

- Conta quantas tuplas foram recuperadas em uma determinada consulta;
- Pode ser aplicada a qualquer consulta SQL;
- O seu resultado é sempre um número inteiro.

Consultas

■ Funções agregadas:

- As principais funções são:

■ COUNT:

- **Exemplo 23:** Verifique quantos empregados trabalham no departamento 2;
 - **SELECT COUNT(*)** AS TotalDeEmpregados
 - **FROM** Empregado **WHERE** CodDepartamento =2;
- O resultado é uma tabela que contém um atributo TotalDeEmpregados e uma única tupla com o valor 2;

Consultas

■ Funções agregadas:

- As principais funções são:

■ COUNT:

- Podemos usar a cláusula COUNT em consultas aninhadas;
- **Exemplo 24:** Recupere o nome de todos os empregados que possuem mais de um dependente;

SELECT Nome **FROM** Empregado E

WHERE

(**SELECT COUNT**(*) **FROM** Dependente D **WHERE**
E.Matricula=D.Empregado) >1

Consultas

- **Funções agregadas:**

- As principais funções são:

- **COUNT:**

- **Exemplo 24:**

- Obtemos a seguinte resposta;

Nome
Maria

Consultas

■ Funções agregadas:

- As principais funções são:

■ MIN e MAX:

- Recuperam, respectivamente, o menor e o maior valor de uma coluna;
- Podem ser aplicados a qualquer tipo de dado que tenha a noção de ordenação;
 - Números, cadeia de caracteres, datas, etc;

■ AVG:

- Recupera a média dos valores de uma determinada coluna;
- Aplicado apenas a tipos numéricos;

Consultas

- **Funções Agregadas:**

- **Exemplo 25:**

- Recupere o menor salário pago pela empresa, o maior salário pela empresa e média salarial dos funcionários;
 - **SELECT MIN**(Salario) **AS** MenorSalario, **MAX**(Salario) **AS** MaiorSalario, **AVG**(Salario) **AS** MediaSalarial **FROM** Empregado;
 - Resultado:

<u>MenorSalario</u>	MaiorSalario	MediaSalarial
2500	4500	3500

Consultas

■ Agrupamento:

- Muitas vezes, precisamos aplicar as funções agregadas para subgrupos de tuplas da relação:
 - Qual a média salarial de um departamento?
 - Quantos funcionários trabalham em cada projeto da empresa?
- Podemos agrupar os elementos de uma relação através da cláusula GROUP BY;
- A cláusula é seguida pelo nome do atributo usado para fazer o agrupamento;
- Podemos selecionar apenas o atributo que é critério de agrupamento e as funções agregadas.

Consultas

■ Agrupamento:

- **Exemplo 26:** Para cada departamento da empresa, recupere o seu código, o seu total de funcionários e a sua média salarial;

```
SELECT CodDepartamento, COUNT(*) AS TotalDeEmpregados, AVG(Salario)  
AS MediaSalarial  
FROM Empregado  
GROUP BY CodDepartamento
```

Consultas

- **Agrupamento:**

- **Exemplo 26:**

- Obtemos a seguinte resposta:

<u>CodDepartamento</u>	TotalDeEmpregados	MediaSalarial
1	2	3500
2	2	3500

Consultas

■ Agrupamento:

- Para recuperarmos atributos adicionais, podemos incluir os atributos adicionais desejados na cláusula GROUP BY após o atributo de agrupamento;
- **Exemplo 27:** Recupere o código de cada projeto, o nome de cada projeto, o total de pessoas que trabalha em cada projeto, e o total de horas trabalhadas por todas as pessoas neste projeto;

```
SELECT P.CodProjeto, P.Nome, COUNT(*) AS NumeroDePessoas, SUM(NumHoras) AS  
HorasTrabalhadas
```

```
FROM TrabalhaProjeto TP, Projeto P
```

```
WHERE TP.CodProjeto = P.CodProjeto
```

```
GROUP BY P.CodProjeto,P.Nome
```

- Neste caso, o agrupamento só é feito após a junção das tabelas.

Consultas

■ Agrupamento:

- **Exemplo 27** (Resposta):
- Obtemos a seguinte resposta:

<u>CodProjeto</u>	Nome	NumeroDePessoas	HorasTrabalhadas
1	Venda Fácil	1	12
2	Max Lucro	3	36
3	Cliente Ouro	2	24

Consultas

■ Agrupamento:

- As vezes não queremos recuperar todos os grupos gerados em um agrupamento;
- Podemos selecionar apenas os grupos que satisfazem algum critério de seleção;
- Podemos fazer esta seleção usando a cláusula HAVING, seguida do critério de seleção desejado;
- Apenas os grupos que satisfazem o critério de seleção são mostrados no resultado da consulta.

Consultas

■ Agrupamento:

- **Exemplo 28:** Recupere o código e o nome dos projetos em que trabalham mais de um empregados;

```
SELECT P.CodProjeto, P.Nome  
FROM Projeto P, TrabalhaProjeto TP  
WHERE P.CodProjeto=TP.CodProjeto  
GROUP BY P.CodProjeto,P.Nome  
HAVING COUNT(*)>1
```

Consultas

- **Agrupamento:**

- **Exemplo 28:**

- Obtemos a seguinte resposta:

<u>CodProjeto</u>	Nome
2	Max Lucro
3	Cliente Ouro

Consultas

- Criando novas colunas em uma consulta:
 - Podemos criar novas colunas em nossa consulta através do comando CASE;
 - O valor para este atributo em cada tupla é calculado no momento da execução da consulta, através de uma ou mais condições;
 - Tuplas que não satisfazem à nenhuma das condições recebem valores nulos;
 - Uma cláusula ELSE, que é opcional, pode ser usada para calcular o valor destas tuplas;

Consultas

- Criando novas colunas em uma consulta:
 - Sintaxe do comando CASE;

CASE

WHEN <cond1> **THEN** <valor1>

WHEN <cond2> **THEN** <valor2>

...

WHEN <condN> **THEN** <valorN>

ELSE <valorDefault>

END

Consultas

- Criando novas colunas em uma consulta:

- Exemplo 29:

SELECT Matricula, Nome, Salario,

CASE

WHEN Salario<3000 **THEN** 'Ganha pouco'

WHEN Salario>4000 **THEN** 'Ganha muito'

END AS Situacao **FROM** Empregado

Consultas

- Criando novas colunas em uma consulta:
 - **Exemplo 29**(Resultado):

Matricula	Nome	Salario	Situacao
1111-1	João	2500	Ganha pouco
1111-2	Maria	2500	Ganha pouco
1111-3	Carlos	4500	Ganha muito
1111-4	Joaquim	4500	Ganha muito



Aula 10 - Consultas



Dúvidas?
vitoria@crateus.ufc.br