



# Revisão - Linguagem SQL

Crícia Felício

# SQL - Sublinguagens

- Pode ser dividida em:
  - Data Definition Language (DDL) – Comandos definem estrutura ou esquema do banco de dados.
    - Principais comandos:
      - CREATE – Criação de banco de dados/tabelas/visões, etc
      - ALTER – Alteração da estrutura das tabelas/banco de dados, visões, etc
      - DROP – deleta uma tabela/banco de dados/visões, etc

# SQL - Sublinguagens

- Data Manipulation Language (DML) – Comandos que fazem o gerenciamento dos dados da base de dados
  - Principais comandos:
    - SELECT : Seleciona dados em uma ou mais tabelas/visões
    - INSERT: Insere dados em uma tabela/visão
    - UPDATE: Atualiza dados de uma tabela/visão
    - DELETE: Apaga dados de uma tabela/visão
    - CALL: Faz a chamada de um procedimento

# SQL - Sublinguagens

- **Data Control Language (DCL)**
  - GRANT: Concessão de privilégios de acesso
  - REVOKE: Retirada de privilégios de acesso
- **Transaction Control (TCL) – Comandos usados para gerenciar as mudanças feitas por comandos DML.**
  - **Permitem que comandos sejam agrupados em transações lógicas**
    - COMMIT: Confirma as alterações realizadas pela transação
    - SAVEPOINT: Cria pontos de controle
    - ROLLBACK: Desfaz as alterações realizadas pela transação

# Acesso ao Mysql

- Mysql comand line client ou Mysql Worbench
- Comandos
  - `show databases;`
    - Mostra os bancos de dados já criados
  - `use nome_do_banco;`
    - Seleciona um banco de dados para ser utilizado
  - `Show tables;`
    - Mostra as tabelas do banco de dados

# Linguagem DDL

- Create database **nome\_do\_banco**;
  - Cria um banco de dados com o nome informado
- Create table **nome\_da\_tabela**(  
Nome\_campo1 tipo,  
Nome\_campo 2 tipo,  
.  
.  
);

# Tipos de Dados do Mysql – Tipos Texto

<b>Tipo</b>	<b>Tamanho</b>	<b>Observação</b>
char(n)	variável	até 255 caracteres. Utiliza os n bytes, mesmo que o tamanho do campo seja inferior.
varchar(n)	variável	até 255 caracteres, utiliza 1 byte para guardar o tamanho e ocupa somente o tamanho do campo.
text	64 Kbytes	até 65.535 caracteres.
TINYTEXT	255 bytes	até 255 caracteres.
MEDIUMTEXT	16 Mbytes	até 16.777.215 caracteres.
LONGTEXT	4 Gbytes	até 4.294.967.295 caracteres.

# Diferença entre char e varchar

Valor	<b>CHAR(4)</b>	Exigência p/ armazenamento	<b>VARCHAR(4)</b>	Exigência p/ armazenamento
' '	'   '	4 bytes	' '	1 byte
'ab'	'ab  '	4 bytes	'ab'	3 bytes
'abcd'	'abcd'	4 bytes	'abcd'	5 bytes
'abcdefgh'	'abcd'	4 bytes	'abcd'	5 bytes



# Tipos de Dados do Mysql – Tipos Numéricos

Tipo	Tamanho	Observação
TINYINT	1 byte	-128 a 127 normal. 0 a 255 UNSIGNED.
SMALLINT	2 bytes	-32768 a 32767 normal. 0 a 65535 UNSIGNED.
MEDIUMINT	3 bytes	-8388608 a 8388607 normal. 0 a 16777215 UNSIGNED.
INT	4 bytes	-2147483648 a 2147483647 normal. 0 a 4294967295 UNSIGNED
BIGINT	8 bytes	-9223372036854775808 a 9223372036854775807 normal. 0 a 18446744073709551615 UNSIGNED
FLOAT	4 bytes	Numero decimal
NUMERIC(SIZE,D)	variável	Size+2 bytes se $D > 0$ , Size+1 bytes se $D = 0$

# Tipos de Dados do Mysql – Tipos Data e Hora

<b>Tipo</b>	<b>Tamanho</b>	<b>Observação</b>
DATE	3 Bytes	Formato : YYYY-MM-DD. Suporta valores entre '1000-01-01' a '9999-12-31'.
DATETIME	8 bytes	Formato: YYYY-MM-DD HH:MI:SS . Suporta valores entre 1000-01-01 00:00:00' to '9999-12-31 23:59:59'
TIME	3 bytes	Formato: HH:MI:SS

# Atributos

- Além do tipo do campo, podem ser utilizados alguns atributos na criação da tabela
  - Auto\_Increment: A coluna terá com valor um número sequencial positivo, que é incrementado a cada nova inserção
  - Not null: Colunas de preenchimento obrigatório
  - Primary Key: indica que o campo é chave primária da tabela

# Atributos

- Há duas maneiras de definir a chave primária
  - Nome\_campo tipo primary key ou
  - Primary key(nome\_campo)
- Foreign key: utilizada para definir uma chave estrangeira
  - Sintaxe:
    - Foreign key(nome\_do\_campo\_tabela\_atual) references nome\_tabela\_origem(nome\_do\_campo\_tabela\_origem)

**Curso**

Cod_curso	Nome	Sigla
1	Sistemas para Internet	TSPI
2	Licenciatura em Computação	LCOMP

**Aluno**

Mat	Nome	Entrada	Cod_curso
1001	Paulo Silva	2016_1	1
1002	Carla Marins	2016_1	1
1003	Marcos Ferreira	2017_1	2

**Disciplina**

Cod_disc	Nome	Sigla	Carga_hor
1	Lógica	Log	105
2	Algoritmos	Alg	80
3	Banco de Dados 1	BD1	75
4	Programação Orientada a Objetos	POO	120

**Professor**

Cod_prof	Nome
1	Clarimundo
2	Mateus
3	Crícia

**Historico**

Cod_turma	Mat	Cod_disc	Media
44	1001	1	6,0
46	1001	2	5,5
47	1001	3	7,0

**Turma**

Cod_turma	Ano	Sem
44	2016	1
46	2016	1
47	2016	2
48	2017	1

**Alocação**

Cod_turma	Cod_disc	Cod_prof
44	1	1
46	3	2
46	2	1
47	3	3
48	4	1

**Pré\_req**

Cod_disc	Cod_disc_req
2	1
4	2

# Exemplo

```
create database academico;  
use academico;
```

```
CREATE TABLE Curso (  
  cod_curso int PRIMARY KEY,  
  nome_curso varchar(40),  
  sigla varchar(8)  
);
```

```
CREATE TABLE Aluno (  
  matr numeric(4) PRIMARY KEY,  
  nome varchar(50),  
  entrada char(6),  
  cod_curso int,  
  FOREIGN KEY(cod_curso) REFERENCES Curso (cod_curso)  
);
```

# Linguagem DDL

- Comando drop
  - Apaga a estrutura de um banco, tabela, visão, procedimentos, etc.
- Drop database **nome\_do\_banco**
  - Apaga o banco de dados
    - Exemplo:  
drop database academico;
- Drop table **nome\_da\_tabela**
  - Apaga a tabela
    - Exemplo:  
-- apaga a tabela aluno  
drop table aluno;

# Linguagem DDL

- \* Comando ALTER TABLE

- \* Altera a estrutura de uma tabela

- Alter table **nome\_tabela** *ação*

- Adicionar uma nova coluna

- ALTER TABLE **nome\_tabela** ADD **nome\_coluna** *tipo*;

- Exemplo:

- ALTER TABLE ALUNO ADD **DATA\_NASC** DATE;



# Linguagem DDL

- Deletar uma coluna

- ALTER TABLE *nome\_da\_tabela*  
DROP COLUMN *nome\_coluna*;

- Exemplo:

```
ALTER TABLE ALUNO DROP COLUMN DATA_NASC;
```

- Alterar o tipo de dado da coluna

- ALTER TABLE *nome\_da\_tabela*  
MODIFY COLUMN *nome\_coluna* *novo\_tipo*;

- Exemplo:

```
ALTER TABLE ALUNO MODIFY COLUMN MATR  
NUMERIC(5);
```

# Linguagem DDL

- Alterar o nome da coluna

- ALTER TABLE nome\_da\_tabela  
CHANGE COLUMN nome\_antigo novo\_nome *tipo*;

- \* Exemplo:

```
ALTER TABLE ALUNO CHANGE COLUMN NOME  
Nome_aluno varchar(50);
```

# Linguagem DDL

- Adicionar uma chave primaria

- ALTER TABLE nome\_da\_tabela  
ADD PRIMARY KEY(nome\_coluna);

- Exemplo: ALTER TABLE aluno ADD PRIMARY KEY(matr);

- Retirar a chave primária

- ALTER TABLE nome\_da\_tabela  
DROP PRIMARY KEY;

- Exemplo: ALTER TABLE aluno DROP PRIMARY KEY;

# Linguagem DDL

- Adicionar uma chave estrangeira
  - ALTER TABLE **nome\_da\_tabela**  
ADD FOREIGN KEY(**nome\_coluna\_tabela\_atual**)  
REFERENCES  
**nome\_tabela\_origem(nome\_coluna\_tabela\_origem);**
    - Exemplo: ALTER TABLE aluno ADD FOREIGN KEY(cod\_curso)  
references curso(cod\_curso);

# Linguagem DML

- Comando INSERT

- Sintaxe:

INSERT INTO nome\_tabela(*campos que serao inseridos, separados por virgula*) values (*valores que serao inseridos*)

Obs: Valores do tipo char, varchar, text, date e time devem vir entre aspas

# Linguagem DML

- \* Comando INSERT

- Exemplo:

- seleciona o banco de dados para uso

```
insert into curso(cod_curso,nome_curso,sigla_curso)
values(1, "Sistemas para Internet","TSPI"),
(2, "Licenciatura em Computação","LCOMP");
select *
from Curso;
```

- OU

```
insert into curso(cod_curso,nome,sigla)
values(1, "Sistemas para Internet","TSPI");
insert into curso(cod_curso,nome,sigla) values
(2, "Licenciatura em Computação","LCOMP");
```

# Exercício

- De acordo com o esquema abaixo, crie o banco de dados empresa e suas tabelas
- Preencha as tabelas de acordo com as informações presentes na figura

- **Esquema empresa**

Cargo(cod\_cargo,nome,nivel,salario)

Departamento(cod\_depto,nome,sigla)

Funcionario(cod\_func,nome,data\_adm,sexo,cod\_cargo,cod\_depto)

Cod\_cargo referencia cargo

Cod\_depto referencia departamento

Cargo				Departamento		
Cod_cargo	Nome	Nível	Salario	Cod_depto	Nome	Sigla
1	Analista de Sistemas	JR	1500	1	Informática	INF
2	Desenvolvedor	JR	2100	2	Financeiro	FIN
3	Desenvolvedor	Pleno	3200	3	Pessoal	RH
4	Atendente	NA	980			
5	Contador	NA	4500			

### Funcionário

Cod_func	Nome	Data_adm	Sexo	Cod_cargo	Cod_depto
1	João Nogueira	12/03/2008	M	1	1
2	Maria Silveira	20/03/2008	F	4	3
3	Marcos Silva	05/07/2008	M	2	1
4	Gabriel Pereira	10/07/2008	M	5	2
5	Carla Junqueira	15/08/2008	F	1	1
6	Janete Rosa	01/10/2008	F	4	3
7	Fernando Silva	03/02/2009	M	3	1
8	Marília Vieira	05/02/2009	F	2	1
9	Patrícia Chaves	01/03/2009	F	5	2
10	João Marques	15/03/2008	M	3	1



# Linguagem DML

- Comando SELECT

- Sintaxe Comando Básico:

SELECT *informações que devem ser selecionadas*

FROM *nome\_tabela1,nome\_tabela2 ....*

WHERE *condicoes*

- Select corresponde a operação de projeção da Álgebra relacional

- Informa quais os as colunas devem ser listadas no resultado da consulta

- Select \*

- Lista todas as informações das tuplas obtidas como resultado da consulta

# Comando SELECT

- Exemplo

- Selecionar todas as informações dos funcionarios

Select \*

from funcionario;

- Selecionar o nome e sexo dos funcionarios

Select nome,sexo

from funcionario;

# Comando SELECT

- Recursos da cláusula where
  - Pode conter diversos operadores combinados através de uma expressão booleana
- Operadores de comparação
  - = : Igualdade
  - <>: Diferente, também ser representado por !=
  - > : Maior
  - < : Menor
  - >=: Maior Igual
  - <=: Menor Igual

# Comando SELECT

- Conectores Lógicos

- And

- C1 And C2 é verdadeiro se C1 é verdadeiro e C2 é verdadeiro

- OR

- C1 or C2 é verdadeiro se C1 e C2 for verdadeiro ou se uma das condições for verdadeira

- Not

- Not C1 é verdadeiro se C1 for falso e é falso se C1 for verdadeiro

# Exemplo

- Selecionar o código de cargo e nome dos funcionários do sexo masculino

```
select nome,cod_cargo  
From funcionario  
where sexo='M';
```

- Selecionar o código de funcionário e nome das mulheres que trabalham no departamento 1

```
Select cod_func,nome  
From funcionario  
where sexo='F' and cod_depto=1;
```

# Exercicio

- Selecione o código do cargo, código do departamento e nome das funcionárias que foram admitidas no ano de 2008
- Liste o nome dos cargos que tem salário entre 2000 e 5000
- Liste o nome e o salario dos cargos que ganham acima de R\$ 3000
- Liste as informações do funcionário Fernando Silva

# Comando SELECT

- Operador between
  - 'A between B and C' é equivalente a:
    - $(A \geq B) \text{ and } (A \leq C)$
  - 'not (A between B and C)' é equivalente a:
    - A not between B and C
  - Exemplo:  
Select nome,cod\_cargo  
From funcionario  
Where data\_adm between '2009-01-01' and '2009-12-31';

# Comando SELECT

- Operador Like e not like
  - É utilizado na cláusula where
    - Faz a busca por parte do conteúdo de uma string
    - Utiliza
      - % - corresponde a uma sequência qualquer de 0 ou mais caracteres.
      - \_ - corresponde a qualquer caracter
- Exemplo:
  - Procure pelos funcionários que tenham o nome ou segundo nome terminado em 'o '

```
Select * from funcionario  
Where nome like '%o %';
```



# Comando SELECT

- Selecione as informações dos funcionários cujo nome começa com a letra J  
Select \* from funcionario  
Where nome like 'J%';
- Liste as informações dos funcionários que possuem nome com 13 caracteres:  
Select \* from funcionario  
Where nome like '\_\_\_\_\_';
- Liste as informações dos cargos cujo nome não terminam com 'dor':  
Select \* from cargo  
Where nome not like '%dor';

# Exercícios

- Liste as informações dos funcionários que tem o nome terminado em 'eira'
- Listar os dados dos funcionários que tem 'Silv' como parte do sobrenome
- Lista o nome e data de admissão dos funcionários que possuem nomes que começam com a letra P ou a letra M.
- Liste as informações dos funcionarios que possuem nomes que não começam com J e nem com G

# Comando SELECT

- Operador IN e Not IN

- Utilizado para verificar se um valor esta(ou não) contido num conjunto de constantes

- Indicado para fazer a comparação com mais de um valor

- Exemplo

Select \*

from funcionario

Where cod\_depto in (2,3);

Irá seleccionar os dados dos funcionários que trabalham no departamento 2 ou 3 equivalente a :

Select \*

from funcionario

Where cod\_depto=2 or cod\_depto=3 ;

# Comando SELECT

- Listar o nome e data de admissão dos funcionários que não são sejam do cargo de código 1 e nem 2

```
Select nome,data_adm  
from funcionario  
Where cod_cargo not in (1,2);
```

# Exercícios

- Utilize operador IN e NOT IN
  - Listar o código e sigla dos departamentos de informática e financeiro
  - Listar o código e o nome dos funcionários que não são mulheres e nem fazem parte dos departamentos 1 e 3
  - Listar o código e os salários dos cargos de atendente, contador e desenvolvedor

# Comando SELECT

- Operador is null e is not null
  - Usado para selecionar dados que contém(ou não contém) o valor null
  - Exemplo:
    - Selecionar as informações dos funcionarios que não possuem a data de admissão preenchida  
Select \*  
From Funcionario  
Where data\_adm is null;
    - Selecionar as informações dos funcionarios que possuem o código do departamento preenchido  
Select \*  
From Funcionario  
Where cod\_depto is not null;

# Comando Select

- \* Operador Like com datas

- Retorna os funcionarios que foram contratados no ano de 2008

```
Select * from funcionario  
where data_adm like '2008-%';
```

- Retorna os funcionarios que foram contratados no mês de março

```
Select * from funcionario  
where data_adm like '%-03-%';
```

# Comando Select

\* Funções year, month e day

```
Select * from funcionario
```

```
where year(data_adm)=2009;
```

```
Select * from funcionario
```

```
where month(data_adm)=3;
```

```
Select * from funcionario
```

```
Where day(data_adm)=10;
```



# Junção de Tabelas e Funções Agregadas

Crícia Felício

# Como fazer consultas que envolvam dados de duas ou mais tabelas?

- \* Junção na cláusula WHERE
  - \* Informar as tabelas envolvidas na cláusula FROM separadas por vírgula
  - \* Informar os atributos de junção na cláusula where
- \* Junção com o operador INNER JOIN
  - \* Informar as tabelas envolvidas na cláusula FROM ligada pelo operador INNER JOIN
  - \* Informar os atributos de junção na cláusula FROM após o nome das tabelas
    - \* Teste de junção é precedido pela cláusula ON
- \* Alias de tabelas (apelido)
  - \* Podem ser utilizados para referenciar uma tabela por outro nome

Cargo				Departamento		
Cod_cargo	Nome	Nível	Salário	Cod_depto	Nome	Sigla
1	Analista de Sistemas	JR	1500	1	Informática	INF
2	Desenvolvedor	JR	2100	2	Financeiro	FIN
3	Desenvolvedor	Pleno	3200	3	Pessoal	RH
4	Atendente	NA	980			
5	Contador	NA	4500			

Funcionário

Cod_func	Nome	Data_adm	Sexo	Cod_cargo	Cod_depto
1	João Nogueira	12/03/2008	M	1	1
2	Maria Silveira	20/03/2008	F	4	3
3	Marcos Silva	05/07/2008	M	2	1
4	Gabriel Pereira	10/07/2008	M	5	2
5	Carla Junqueira	15/08/2008	F	1	1
6	Janete Rosa	01/10/2008	F	4	3
7	Fernando Silva	03/02/2009	M	3	1
8	Marília Vieira	05/02/2009	F	2	1
9	Patrícia Chaves	01/03/2009	F	5	2
10	João Marques	15/03/2008	M	3	1

Cargo

Departamento

Funcionário



# Exemplo de Junção na Cláusula Where

Listar os nomes dos funcionários que trabalham no departamento de informática

Select

    f.nome

From

    funcionario f,

    departamento d

Where

    f.cod\_depto = d.cod\_depto

    and d.nome = 'Informática';

# Exemplo de Junção com Operador INNER JOIN

Listar os nomes dos funcionários que trabalham no departamento de informática

Select

f.nome

From

funcionario f INNER JOIN departamento d

ON f.cod\_depto = d.cod\_depto

Where d.nome = 'Informática';

# Exemplo de Junção

\* Com 3 tabelas

Listar o nome e data de admissão do funcionário juntamente com seu cargo e nome do departamento

Select

f.nome, f.data\_adm, c.nome nome\_cargo, d.nome nome\_depto

From

funcionario f,

cargo c,

departamento d

Where

f.cod\_depto = d.cod\_depto

and f.cod\_cargo = c.cod\_cargo;

# Exemplo de Junção

\* Com 3 tabelas

Listar o nome e data de admissão do funcionário juntamente com seu cargo e nome do departamento

Select

f.nome, f.data\_adm, c.nome nome\_cargo, d.nome  
nome\_depto

From

funcionario f **INNER JOIN** cargo c **ON** f.cod\_cargo =  
c.cod\_cargo

**INNER JOIN** departamento d **ON** f.cod\_depto = d.cod\_depto;



# Exercícios

- \* Selecionar o nome e o sexo dos funcionários que são desenvolvedores.
- \* Liste o nome do funcionário e a sigla do departamento de cada funcionário.
- \* Liste o nome, sexo e salário de cada funcionario contratado no ano de 2009.
- \* Liste o nome, data de admissão e nome do departamento dos funcionários do sexo feminino.
- \* Liste os dados dos funcionários que trabalham no departamento de informática ou financeiro.

# Exercícios URI

- \* 2605
- \* 2606
- \* 2609
- \* 2612
- \* 2613
- \* 2614
- \* 2616
- \* 2617
- \* 2618
- \* 2619
- \* 2620
- \* 2621
- \* 2622
- \* 2623
- \* 2742

# Funções Agregadas

# Funções Agregadas

- \* São funções pré-definidas do SQL que pode ser utilizadas nas consultas
  - \* Min(x), retorna o mínimo valor de uma coluna x
  - \* Max(x), retorna o máximo valor de uma coluna x
  - \* Count(x), retorna a quantidade de valores de uma coluna x
  - \* Count(\*), retorna a quantidade de linhas da relação
  - \* Sum(x), retorna a soma de valores de uma coluna x
  - \* Avg(x), retorna a média de valores de uma coluna x

# Funções Agregadas

- \* No BD empresa
  - \* Listar o menor salario de todos os cargos  
`Select min(salario)`  
`From cargo;`
  - \* Listar a data mais recente de admissão de um funcionário  
`Select max(data_adm)`  
`From funcionario;`
  - \* Listar a quantidade funcionários do sexo masculino  
`Select count(*)`  
`From funcionario`  
`Where sexo='M';`

# Funções Agregadas

- \* Listar a média de salários para os cargos de nível JR

```
Select avg(salario)
```

```
From cargo
```

```
Where nivel='JR';
```

# Funções Agregadas

\* Listar o total pago em salários para Desenvolvedores

```
Select sum(c.salario)
```

```
From cargo c inner join funcionario f ON
```

```
c.cod_cargo=f.cod_cargo
```

```
Where c.nome='Desenvolvedor' ;
```

# Exercícios

- \* Selecione o maior salário do departamento Pessoal
- \* Listar o total pago em salários para funcionários do departamento financeiro
- \* Listar a quantidade de funcionários do departamento de informática
- \* Listar o menor salário de funcionários do departamento de informática



# Exercícios

- \* Selecione o maior salário, menor salário, média e soma dos salários dos funcionários do departamento de Informática
- \* Listar a quantidade de funcionárias do sexo feminino que trabalham no departamento Financeiro ou Pessoal
- \* Selecionar a média dos salários dos funcionários contratados no primeiro semestre do ano de 2008
- \* Listar a quantidade de funcionários que ganham abaixo de R\$ 2000

# Cláusulas ORDER BY, GROUP BY e Having

Crícia Felício

# Comando Distinct

- \* Elimina duplicatas e retorna somente os valores distintos para uma coluna
  - \* A consulta abaixo irá selecionar somente os valores distintos para nome de cargo que tiveram funcionários admitidos no ano de 2008 ou 2009

```
select distinct c.nome  
from funcionario f inner join cargo c on  
f.cod_cargo=c.cod_cargo  
where (year(f.data_adm)=2008 or year(f.data_adm)=2009);
```

# Cláusula ORDER BY

# Cláusula ORDER BY

- \* Utilizada para ordenar as linhas de acordo com a informação de uma ou mais colunas
- \* Deve vir no final da consulta
- \* Pode ser ordenado de forma Ascendente ou Descendente
  - \* *Default* Ascendente

Exemplo: Listar as informações dos funcionários, ordenados pelo campo nome

```
Select *  
From funcionario  
Order by nome;
```

# Cláusula ORDER BY

Exemplo: Listar as informações dos funcionários,  
ordenados em ordem decrescente pelo campo nome

```
Select *
```

```
From funcionario
```

```
Order by nome desc;
```

# Cláusula Order By

- \* Listar o nome do cargo e o nome do funcionário ordenados pelo cargo e nome do funcionário

```
select f.nome nome_func, c.nome nome_cargo  
from funcionario f inner join cargo c on  
f.cod_cargo=c.cod_cargo  
order by c.nome, f.nome;
```

# Cláusula Order By

- \* Listar o nome do cargo e o nome do funcionário ordenados pelo cargo e nome do funcionário

```
select f.nome nome_func,c.nome nome_cargo  
from funcionario f inner join cargo c on  
f.cod_cargo=c.cod_cargo  
order by nome_cargo,nome_func;
```

Pode ser usado o alias (apelido) da coluna



# Cláusula Order By

- \* Listar o nome do cargo e o nome do funcionário ordenados pelo cargo e nome do funcionário

```
select f.nome nome_func,c.nome nome_cargo  
from funcionario f inner join cargo c on  
f.cod_cargo=c.cod_cargo  
order by 2,1;
```

Pode ser usado o índice do campo no resultado

# Cláusula Group BY


# Cláusula Group By

- \* Como listar a quantidade de funcionários por departamento?
- \* Como listar os valores pagos em salários por departamento?
- \* Como apresentar em uma única consulta a quantidade de funcionários de cada sexo?
- \* Cláusula Group BY
  - \* Significado de Group By: “Agrupado por

# Cláusula Group BY - Funcionamento

Selecionar os dados dos funcionários ordenados pelo código do departamento

cod_func	nome	data_adm	sexo	cod_cargo	cod_depto
1	João Nogueira	2008-03-12	M	1	1
8	Marília Vieira	2009-02-05	F	2	1
7	Fernando Silva	2009-02-03	M	3	1
10	João Marques	2008-03-15	M	3	1
5	Carla Junqueira	2008-08-15	F	1	1
3	Marcos Silva	2008-07-05	M	2	1
4	Gabriel Pereira	2008-07-10	M	5	2
9	Patrícia Chaves	2009-03-01	F	5	2
2	Maria Silveira	2008-03-20	F	4	3
6	Janete Rosa	2008-10-01	F	4	3



# Cláusula Group BY

- \* Possui como argumento o nome ou referência de um ou mais atributos
- \* Faz a seleção de linhas de uma tabela em grupos
  - \* Onde cada grupo possui colunas com o mesmo valor
- \* GROUP BY + funções agregadas

# Cláusula Group By

- \* Exemplo: Listar a quantidade de funcionários por código de departamento

```
select cod_depto,count(*) qtde_func  
from funcionario  
group by cod_depto ;
```

Ou

```
select cod_depto,count(*) qtde_func  
from funcionario  
group by 1 ; -- agrupa pela primeira coluna do resultado
```

# Cláusula Group BY

- \* Pode ser usado mais de um critério de agrupamento
  - \* Exemplo: Listar a quantidade de funcionários por código de departamento e por código de cargo
- ```
select cod_depto,cod_cargo,count(*) qtde_func  
from funcionario  
group by cod_depto,cod_cargo;
```

# Cláusula Group BY

- \* Listar o ano e número de funcionários contratados em cada ano

```
Select year(data_adm) ano,count(*) qtde_func_adm  
From funcionario  
Group by ano;
```



# Cláusula Group By

- \* Listar o nome do departamento e a média de salários para cada departamento

```
Select d.nome,avg(c.salario) media_sal  
From funcionario f inner join cargo c on  
c.cod_cargo=f.cod_cargo  
inner join departamento d on  
d.cod_depto=f.cod_depto  
Group by d.nome;
```

# Exercícios

- \* Listar o menor salário e maior salário por nome de departamento, ordenando o resultado pelo nome do departamento.
  - \* A cláusula order by deve vir após a cláusula group by
- \* Listar o ano, mês e quantidade de funcionários contratados agrupados pelo ano e o mês
- \* Listar para cada nome de cargo a quantidade de níveis cadastrados ordenando o resultado pelo nome do cargo
  - Listar o valor pago em salários por nome de departamento ordenando o resultado em ordem decrescente pelo valor pago
  - Listar a menor e a maior data de contratação de cada departamento

# Exercícios

- \* Listar para cada nome de departamento, a quantidade de funcionários do sexo feminino e a quantidade de funcionários do sexo masculino
- \* Listar o nome do cargo e total pago em salários para cada cargo ordenando o resultado pelo nome do cargo
- \* Para funcionários que trabalham no departamento de informática, listar o nome do cargo, nível e total pago em salários para cada cargo e nível

# Cláusula Having

# Cláusula Having

- \* Como adicionar uma condição ao agrupamento?
  - \* Exemplo: Listar para cada nome de cargo, **que possui mais de um nível**, a quantidade de níveis cadastrados.  

```
select nome, count(*) qtde_niveis  
from cargo  
group by nome  
having count(*)>1;
```

# Cláusula Having

- \* Utilizada para especificar uma condição para que uma linha seja incluída no grupo
- \* A condição deve ser aplicada ao grupo através de uma função agregada
- \* Pode-se entender a declaração HAVING como sendo uma cláusula WHERE para a declaração GROUP BY

# Cláusula Having

- \* Para departamento com mais de 2 funcionários, listar o nome do departamento e a quantidade de funcionários do departamento

```
Select d.nome,count(*) qtde_func
```

```
From funcionario f inner join departamento d on  
d.cod_depto=f.cod_depto
```

```
Group by d.nome
```

```
having count(*)>2;
```

# Cláusula Having

- \* Listar o nome do departamento e a soma de salários pagos pelos departamento que possuem média salarial superior a R\$ 1000

```
Select d.nome,sum(c.salario) soma_sal  
From funcionario f inner join cargo c on  
c.cod_cargo=f.cod_cargo  
inner join departamento d on d.cod_depto=f.cod_depto  
Group by d.nome  
having avg(c.salario)>1000;
```



# Função Group\_concat

\* Faz o agrupamento de dados por grupo

Ex. Listar o nome da região e seus estados

```
select e.regiao,group_concat(e.nome)
```

```
from estado e
```

```
group by e.regiao;
```

# Função Group\_concat

| regiao       | group_concat(e.nome)                                                              |
|--------------|-----------------------------------------------------------------------------------|
| Centro-Oeste | Goiás,Mato Grosso,Mato Grosso do Sul,Distrito Federal                             |
| Nordeste     | Paraíba,Piauí,Rio Grande do Norte,Maranhão,Pernambuco,Ceará,Bahia,Sergipe,Alagoas |
| Norte        | Rondônia,Roraima,Acre,Pará,Amazonas,Amapá,Tocantins                               |
| Sudeste      | Minas Gerais,Espírito Santo,Rio de Janeiro,São Paulo                              |
| Sul          | Rio Grande do Sul,Santa Catarina,Paraná                                           |
|              |                                                                                   |
|              |                                                                                   |

# Função Group\_concat

```
select e.nome,group_concat(pe.ano_pib),  
group_concat(pe.valor_pib)  
from estado e, pib_estado pe  
where e.id=pe.id_estado  
group by pe.id_estado;
```

# Função Group\_concat

| nome               | group_concat(pe.ano_pib) | group_concat(pe.valor_pib) |
|--------------------|--------------------------|----------------------------|
| Acre               | 2015,2017                | 13.622,14.271              |
| Alagoas            | 2015,2017                | 46.364,52.843              |
| Amazonas           | 2015,2017                | 86.56,93.204               |
| Amapá              | 2015,2017                | 13.861,15.48               |
| Bahia              | 2015,2017                | 245.025,268.661            |
| Ceará              | 2015,2017                | 130.621,147.89             |
| Distrito Federal   | 2015,2017                | 215.613,244.693            |
| Espírito Santo     | 2015,2017                | 120.363,113.352            |
| Goiás              | 2015,2017                | 173.632,191.899            |
| Maranhão           | 2015,2017                | 78.475,89.524              |
| Minas Gerais       | 2015,2017                | 519.326,576.199            |
| Mato Grosso do Sul | 2015,2017                | 83.082,96.372              |
| Mato Grosso        | 2015,2017                | 107.418,126.805            |
| Pará               | 2015,2017                | 130.803,155.195            |
| Paraíba            | 2015,2017                | 56.14,62.387               |
| Pernambuco         | 2015,2017                | 156.955,181.551            |
| Piauí              | 2015,2017                | 39.148,45.359              |