



SQL

*Structured Query Language*



# Linguagem

- DDL (*Data Definition Language*) :  
linguagem de definição de dados
- DML (*Data Manipulation Language*) :  
linguagem de manipulação de dados



# DDL

- Comandos:
  - CREATE
  - DROP
  - ALTER
  - RENAME
  - TRUNCATE



# Criando Tabelas

```
CREATE TABLE nome_tabela  
    ( coluna    tipo    [constraint coluna],  
      ...  
      ...  
      ...  
    [constraint de tabela] );
```

# Tipo de dados - MySQL

Tipo	Descrição
VARCHAR(size)	Valores de caractere de comprimento variável
CHAR(size)	Valores de caractere de comprimento fixo
INTEGER	Valores Numéricos
DECIMAL(p,s)	Valores Numéricos
DATE	Valores data
BLOB	Valores de caractere de comprimento variável até 2GB
LOB	Caractere de comprimento variável para dados binários

# CREATE TABLE

```
CREATE TABLE pessoa (  
    codigo INTEGER(7)    PRIMARY KEY,  
    fone    VARCHAR(15) NOT NULL,  
    nome    VARCHAR(35) NOT NULL )
```

```
CREATE TABLE pessoa (  
    codigo INTEGER(7) ,  
    fone    VARCHAR(15) NOT NULL,  
    nome    VARCHAR(35) NOT NULL,  
    PRIMARY KEY (codigo) )
```



# Constraints

- NOT NULL
- UNIQUE
- PRIMARY KEY
- FOREIGN KEY
- CHECK

# Constraints: sintaxe

- Constraint no nível de coluna
  - coluna [CONSTRAINT nome\_constraint] tipo\_constraint,
- Constraint no nível de tabela
  - coluna, ...  
[CONSTRAINT nome\_constraint tipo\_constraint (coluna, ...),



# Constraints

```
CREATE TABLE TB_PESSOA(  
    codigo        INTEGER(7) PRIMARY KEY,  
    fone          VARCHAR(15) NOT NULL,  
    nome          VARCHAR(35) NOT NULL,  
    cpf           VARCHAR(13) UNIQUE,  
    casado        INTEGER(7)  
                REFERENCES TB_PESSOA(codigo),  
    sexo          char(1) check (sexo IN ('F','M')) );
```



# Confirmando a criação da Tabela

- Confirme a criação de uma tabela no banco de dados juntamente com seus nomes de coluna usando o comando **DESCRIBE**
  - **DESCRIBE <NOME DA TABELA>**
  - **DESC <NOME DA TABELA>**



# ALTER TABLE

- Altera estrutura e *constraints* de uma tabela
- O comando ALTER TABLE
  - Adicionar e alterar colunas
  - Incluir ou remover *constraints*
  - Habilitar e desabilitar *constraints*

# Comando ALTER

```
ALTER TABLE nome_tabela  
ADD [CONSTRAINT nome_constraint]  
    tipo_constraint (nome_coluna)
```

```
ALTER TABLE <nome_tabela>  
DROP PRIMARY KEY | UNIQUE (nome_coluna) |  
    CONSTRAINT nome_constraint [CASCADE];
```

```
ALTER TABLE nome_tabela  
DROP PRIMARY KEY | UNIQUE (nome_coluna) | CONSTRAINT  
<nome da constraint> [CASCADE];
```

# Comando DROP

- Elimina uma tabela
  - Todos os dados e estruturas da tabela são excluídos
  - Todas as transações pendentes serão efetivadas (commit)
  - Todos índices serão eliminados

```
DROP TABLE <nome da tabela>  
[CASCADE CONSTRAINTS];
```

# Comando RENAME

- Altera o nome de um objeto
  - Deve ser o proprietário do objeto: tabela, view, seqüência, sinônimo...

```
RENAME tipo_objeto objeto_antigo TO  
objeto_novo;
```

# Comando TRUNCATE

- Remove todas as linhas de uma tabela
  - Libera o espaço de armazenamento utilizado por esta tabela
  - Semelhante a instrução DELETE

```
TRUNCATE TABLE nome_tabela;
```



# SELECT

Comando DML



# SELECT

- **Executa uma consulta no Banco de Dados**

```
SELECT [DISTINCT] (*, coluna [alias])  
FROM tabela  
[WHERE condição]  
[ORDER BY {coluna, exp} [ASC|DESC]];
```

**DISTINCT**      valores distintos (sem duplicação)

**\***                      todas as colunas da tabela

**Alias**                      apelido para a coluna na exibição dos dados

# Exemplos

Seleciona todos os dados de todas as colunas da tabela de Departamento

```
SELECT * FROM Departamento;
```

Seleciona todos os dados da coluna nome na tabela de Departamento

```
SELECT nome FROM Departamento;
```



# Expressões aritmética

- Adição (+)
- Subtração (-)
- Multiplicação (\*)
- Divisão (/)

# Exemplo

Cálculo do salário anual dos empregados

```
SELECT nome, salario*12 "Salario Anual"  
FROM Empregado
```

Nome	Salario Anual
Maria	10000
Jose	5000
João	30000

# Exemplo

Cálculo do salário anual dos empregados

```
SELECT nome, salario*12 + salario + salario/3  as "Salario Anual"  
FROM   Empregado
```

Nome	Salario Anual
Maria	10000
Jose	5000
João	30000



# Concatenação de Colunas

O operador de concatenação é || (duas barras verticais)

```
SELECT Prinome || UltimoNome "Nome do empregado"  
FROM   Empregado
```

Nome	do Empregado
Maria	Silva
Jose	Costa
João	Tornado

# Linhas Duplicadas

Seleciona todos os cargos dos empregados dados de todas as colunas da tabela de Departamento

```
SELECT cargo  
FROM Empregado;
```

```
Cargo  
Programador  
Programador  
Analista  
Analista
```

# Linhas Duplicadas

Seleciona os diferentes tipos de cargos dos empregados

```
SELECT DISTINCT cargo  
FROM Empregado;
```

```
Cargo  
Programador  
Analista
```



# Linhas Duplicadas

Seleciona os diferentes tipos de cargos dos empregados

```
SELECT DISTINCT cargo  
FROM Empregado;
```

```
Cargo  
Programador  
Analista
```

# ORDER BY

Seleciona os diferentes tipos de cargos dos empregados em ordem ascendente

```
SELECT DISTINCT cargo
FROM Empregado
ORDER BY cargo;
```

```
Cargo
Analista
Programador
```

# ORDER BY

Seleciona os diferentes tipos de cargos dos empregados em ordem ascendente

```
SELECT DISTINCT cargo  
FROM Empregado  
ORDER BY 1 DESC;
```

Cargo  
Programador  
Analista

# ORDER BY

Cálculo do salário anual dos empregados

```
SELECT  nome, salario*12 "Salario Anual"  
FROM    Empregado  
ORDER BY 2
```

Nome	Salario Anual
Jose	5000
Maria	10000
João	30000

# ORDER BY

Seleciona os diferentes tipos de cargos dos empregados em ordem ascendente

```
SELECT    nome, departamento, salarioa
FROM      Empregado
ORDER BY  departamento, salario DESC;
```

Nome	Departamento	Salario
Maria	10	1450
Jose	31	1400
Joao	31	750
Fabio	33	1600
Marcos	33	800



# Where

- Seleciona registro com uma determina condição
- Strings e datas devem estar entre aspas simples
- Número não deve estar entre aspas simples
- Valores de caracter são **case-sensitive**

# Where

Seleciona os diferentes tipos de cargos dos empregados em ordem ascendente

```
SELECT nome, cargo  
FROM   Empregado  
WHERE  cargo = 'Analista';
```

Nome	Cargo
João	Analista

# Where

Seleciona os diferentes tipos de cargos dos empregados em ordem ascendente

```
SELECT nome, cargo  
FROM   Empregado  
WHERE  cargo = 'ANALISTA';
```

Nenhuma linha selecionada





# Comparadores e Operadores Lógicas

- **Operadores lógicos de comparação**

= > >= < <=

- **Operadores de comparação SQL**

IN (lista)

LIKE

IS NULL

- **Operadores lógicos**

AND

OR

NOT

# Expressão de Negação

- **Operadores lógicos de comparação**

`!=` `<>` `^=`

- **Operadores de comparação SQL**

`NOT IN (lista)`

`NOT LIKE`

`IS NOT NULL`

# where

Cálculo do salário anual dos empregados

```
SELECT    nome, salario
FROM      Empregado
WHERE     salario > 7000
ORDER BY  2
```

Nome	Salario
Maria	10000
João	30000



# where

Cálculo do salário anual dos empregados

```
SELECT    nome, salario  
FROM      Empregado  
WHERE     salario =10000
```

Nome	Salario
Maria	10000

# AND

Cálculo do salário anual dos empregados

```
SELECT    nome, salario
FROM      Empregado
WHERE     salario > 7000
AND       nome = 'Maria'
```

Nome	Salario
Maria	10000



# OR

```
SELECT      nome, salario  
FROM        Empregado  
WHERE       salario > 7000  
OR          nome = 'Maria'
```

Nome	Salario
Maria	10000
João	30000



# Outros comandos DML



# Comando de manipulação

- INSERT
- UPDATE
- DELETE



# Adicionando novas linhas em uma tabela

```
INSERT INTO tabela [colunas]  
VALUES (valores);
```

Apenas uma linha é inserida por vez com esta sintaxe

# Inserindo novas linhas

departamento

(id	integer(7)	PRIMARY KEY,
nome	varchar(30)	NOT NULL,
regiao	integer (7)	REFERENCES regiao(id));

```
INSERT INTO departamento  
VALUES (1, 'VENDA', 1);
```

Liste os valores na ordem default das colunas na tabela

Coloque valores de caracter e de data entre apóstrofos

# Adicionando valores nulos

```
INSERT INTO departamento (id,nome)  
VALUES (1, 'VENDA');
```

```
INSERT INTO departamento  
VALUES (1, 'VENDA', NULL);
```

```
INSERT INTO departamento  
VALUES (1, 'VENDA', '');
```



# Inserindo linhas de outra tabela

```
INSERT INTO tabela [colunas]  
subquery;
```

Não use a cláusula VALUES

Use na subquery o mesmo número de colunas usadas na cláusula INSERT

# Atualizando linhas em uma tabela

```
UPDATE tabela  
SET      coluna = valor [, coluna=valor]  
[WHERE condição];
```

## Atualizando linhas em uma tabela

```
UPDATE empregado  
SET      NumDep = 7  
WHERE    matrEmp='12345';
```

```
UPDATE      empregado  
SET         NumDep = 7, fone='222-3333'  
WHERE       matrEmp='12345';
```

# Removendo linhas

```
DELETE [FROM]    table  
[WHERE           condição];
```

# Exemplos

Remove todos os registros da tabela empregado

```
DELETE empregado
```

```
DELETE empregado  
WHERE matrEmp='12345';
```





# Removendo Linhas: Erro de Constraint de Integridade

Se você tentar excluir uma linha que contém uma chave primária usada como chave estrangeira em outra tabela, ocorrerá um erro de *constraint* de integridade



# Funções de Grupo (GROUP BY)

- Mostra estatística para diferentes grupos
- Inclui ou exclui registros de grupos usando a clausura HAVING



# GROUP BY e HAVING

```
SELECT coluna,  
      FROM tabela  
      [WHERE condição]  
      [GROUP BY expressão]  
      [HAVING condição do grupo]  
      [ORDER BY colunas]
```



# Funções de Grupo

- AVG – média
- COUNT – contador
- MAX – máximo
- MIN - mínimo




Verifique qual é o maior, o menor, soma e a média salarial da tabela de empregado

```
SELECT AVG(salario) "Média Salarial",  
        SUM(salario) "Soma dos salários"  
        MIN(salario) "Menor Salario"  
        MAX(salario) "Maior Salário"  
FROM empregado
```




Quantos departamento existem na tabela de departamento ?

```
SELECT COUNT(*)  
FROM departamento
```



Quantos empregados do sexo M tem na tabela de empregado ?

```
SELECT count(*)  
FROM empregado  
WHERE sexo = 'M'
```



Qual a média salarial de cada cargo da tabela de empregado ?

```
SELECT AVG(salario) "Média Salarial", cargo  
FROM empregado  
GROUP BY cargo
```





# **SELECT MÚLTIPLAS TABELAS**



# Produto Cartesiano

```
SELECT *  
FROM tabela1, tabela2
```

<b>Matr</b>	<b>Nome</b>	<b>Sexo</b>	<b>Dep</b>
<b>111</b>	<b>PEDRO</b>	<b>M</b>	<b>VE</b>
<b>222</b>	<b>MARIA</b>	<b>F</b>	<b>EN</b>
<b>123</b>	<b>CLAUDIA</b>	<b>F</b>	<b>VE</b>

<b>CodDep</b>	<b>NomeDep</b>
<b>VE</b>	<b>VENDAS</b>
<b>EN</b>	<b>ENGEN.</b>

**SELECT \***  
**FROM empregado, departamento**

	<b>Nome</b>	<b>Sexo</b>	<b>Dep</b>	<b>CodDep</b>	<b>NomeDep</b>
<b>111</b>	<b>PEDRO</b>	<b>M</b>	<b>VE</b>	<b>VE</b>	<b>VENDAS</b>
<b>111</b>	<b>PEDRO</b>	<b>M</b>	<b>VE</b>	<b>EN</b>	<b>ENGEN.</b>
<b>222</b>	<b>MARIA</b>	<b>F</b>	<b>EN</b>	<b>EN</b>	<b>ENGEN.</b>
<b>222</b>	<b>MARIA</b>	<b>F</b>	<b>EN</b>	<b>VE</b>	<b>VENDAS</b>
<b>123</b>	<b>CLAUDIA</b>	<b>F</b>	<b>VE</b>	<b>VE</b>	<b>VENDAS</b>
<b>123</b>	<b>CLAUDIA</b>	<b>F</b>	<b>VE</b>	<b>EN</b>	<b>ENGEN.</b>



# Join


```
SELECT tabela.coluna, tabela.coluna  
FROM tabela1, tabela2  
WHERE tabela1.coluna1 = tabela2.coluna2
```

<b>Matr</b>	<b>Nome</b>	<b>Sexo</b>	<b>Dep</b>
<b>111</b>	<b>PEDRO</b>	<b>M</b>	<b>VE</b>
<b>222</b>	<b>MARIA</b>	<b>F</b>	<b>EN</b>
<b>123</b>	<b>CLAUDIA</b>	<b>F</b>	<b>VE</b>

<b>CodDep</b>	<b>NomeDep</b>
<b>VE</b>	<b>VENDAS</b>
<b>EN</b>	<b>ENGEN.</b>

**SELECT \***  
**FROM empregado, departamento**  
**WHERE empregado.Dep = Departamento.codDep**


<b>Matr</b>	<b>Nome</b>	<b>Sexo</b>	<b>Dep</b>	<b>CodDep</b>	<b>NomeDep</b>
<b>111</b>	<b>PEDRO</b>	<b>M</b>	<b>VE</b>	<b>VE</b>	<b>VENDAS</b>
<b>222</b>	<b>MARIA</b>	<b>F</b>	<b>EN</b>	<b>EN</b>	<b>ENGEN.</b>
<b>123</b>	<b>CLAUDIA</b>	<b>F</b>	<b>VE</b>	<b>VE</b>	<b>VENDAS</b>



Liste o nome do empregado e o nome do departamento onde ele trabalha

```
SELECT empregado.nome, departamento.NomeDep  
FROM empregado, departamento  
WHERE empregado.Dep = departamento.codDep
```

<b>Nome</b>	<b>NomeDep</b>
<b>PEDRO</b>	<b>VENDAS</b>
<b>MARIA</b>	<b>ENGEN.</b>
<b>CLAUDIA</b>	<b>VENDAS</b>



Liste o nome do empregado e o nome do departamento onde ele trabalha. Selecione apenas os empreg. do sexo M

```
SELECT empregado.nome, departamento.NomeDep  
FROM empregado, departamento  
WHERE empregado.Dep = departamento.codDep  
AND empregado.sexo = 'M'
```

Nome	NomeDep
PEDRO	VENDAS



# Join

```
SELECT * FROM table1, table2;
```

```
SELECT * FROM  
    table1 INNER JOIN table2 ON table1.id=table2.id;
```

```
SELECT * FROM  
    table1 LEFT JOIN table2 ON table1.id=table2.id;
```





# Subqueries

Escrever queries aninhadas para o banco de dados.

Uma subquery é um SELECT que contem outros SELECTs

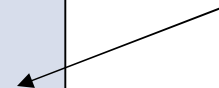
# Subquery

Query  
principal →

```
SELECT ...  
FROM ...  
WHERE ...
```

```
SELECT ...  
FROM ...  
WHERE ...
```

Subquery





# Subqueries (sintaxe)

```
SELECT lista_select  
FROM   tabela  
WHERE expressão operador  
        ( SELECT lista_select  
          FROM   tabela)
```

A subquery é executada antes da query principal



# Subquery

- Uma subquery deve estar sempre entre parênteses
- Uma subquery deve aparecer do lado direito do operador
- Subquery pode ser usado na clausura FROM



# Exemplo

Selecione o nome de todos os empregados que trabalham no mesmo departamento da Maria

Passo 1: Encontrar o departamento da Maria

Passo 2: Encontrar o nome dos empregados do departamento da Maria



# Exemplo

```
SELECT nome  
  FROM empregado  
WHERE dep = ( SELECT dep  
                FROM empregado  
                WHERE nome = 'Maria' )
```



# Exemplo

```
SELECT b.nome  
  FROM empregado a, empregado b  
WHERE a.nome = 'Maria'  
      AND a.dep = b.dep;
```



# Função de grupo em subquery

```
SELECT nome, salario  
FROM empregado  
WHERE salario < ( SELECT AVG(salario)  
                  FROM empregado)
```



# Retornando mais de um valor

```
SELECT nome  
  FROM empregado  
WHERE dep = ( SELECT dep  
                FROM departamento  
              WHERE nome = 'Vendas'  
                OR nome = 'Finanças')
```

# Múltiplos valores

Para trabalhar com múltiplos valores use o operador IN

```
SELECT nome  
  FROM empregado  
WHERE dep IN ( SELECT dep  
                  FROM departamento  
                WHERE nome = 'Vendas'  
                  OR nome = 'Finanças')
```

# Having com subquery

```
SELECT dep_id , AVG(salario)
FROM empregado
GROUP BY dep_id
HAVING  AVG(salario) > ( SELECT AVG(salario)
                        FROM empregado
                        WHERE dep_id = 30)
```