

# Laboratório de Bases de Dados

PROF. JOSÉ FERNANDO RODRIGUES JR

AULA 2 – SQL-DDL

---

MATERIAL ORIGINAL EDITADO: PROFA. ELAINE PARROS MACHADO DE SOUSA

# SQL

- Linguagem declarativa – não procedural
- IBM - década de 70
- Interface entre usuários e o SYSTEM R
- Padrão de mercado
  - Ansi/ISO
  - simplicidade
  - grande poder de consulta

# SQL

---

## Recursos:

- DDL – Linguagem de Definição de Dados
- DML – Linguagem de Manipulação de Dados
- criação de visões (*views*)
- especificações de segurança e autorizações
- definição de restrições de integridade
- controle de transação
- ....

# SQL - Introdução

- O padrão SQL (ISO/ANSI)
  - **SQL 2023**
    - property Graph Queries
  - **SQL 2019**
    - multidimensional arrays
  - **SQL 2016**
    - JSON
  - **SQL 2011**
    - temporal databases
  - **SQL 2008**
    - instead of triggers, case statements
  - **SQL 2006**
    - SQL/XML
  - **SQL 2003**
    - SQL/XML
  - **SQL99 (SQL3)**
    - conceitos de orientação a objetos
  - **padrões anteriores**
    - SQL92 – SQL2
    - SQL86

Na prática, os fabricantes usam suas próprias variações de SQL. Sendo que o padrão serve apenas de referência. Um pouco mais sobre cada SGBD:

<https://www.altexsoft.com/blog/business/comparing-database-management-systems-mysql-postgresql-mssql-server-mongodb-elasticsearch-and-others/> (maio, 2023)

DDL (CREATE, DROP, ALTER)

---

# DDL - Introdução

CREATE, DROP, ALTER

Elementos fundamentais da linguagem

---

- **DATABASE**
- **USER**
- **ROLE**
- **SCHEMA**
- **TABLESPACE**
- **TABLE**
- **INDEX**
- **FUNCTION**
- **SEQUENCE**
- **TRIGGER**
- **VIEW**
- .....

Todos os elementos podem ser criados (**CREATE**), alterados (**ALTER**), ou removidos (**DROP**)

# Comandos DDL

---

**CREATE TABLE** - criar uma tabela, definir colunas e restrições

```
CREATE TABLE tabela (  
    atrib1 tipo [<restrições da coluna 1>],  
    atrib2 tipo [<restrições da coluna 2>],  
    ....  
    atribn tipo [<restrições da coluna n>],  
  
    <restrições da tabela>  
);
```

# CREATE TABLE

---

## Restrições de colunas

- NOT NULL
- DEFAULT *valor*
- CHECK(*condição*)

```
CREATE TABLE tabela (  
    atrib1 tipo [(tamanho)] [NOT NULL | DEFAULT valor]  
        [CHECK (condição)],  
    atrib2 tipo [(tamanho)] [NOT NULL | DEFAULT valor]  
        [CHECK (condição)],  
    . . .
```






# CREATE TABLE

---

## Restrições de tabela

- PRIMARY KEY ( *<atributos chave primária>* )
- UNIQUE ( *<atributos chave candidata>* )
- FOREIGN KEY ( *<atributos chave estrangeira>*  
REFERENCES *tabelaRef* [ (*<chave primária>*) ]  
[*<ações>*]
- *<ações>*
  - ON DELETE | ON UPDATE
    - CASCADE | SET NULL | SET DEFAULT
- CHECK (*condição*)

# SQL – Alguns tipos de dados

- **INTEGER | SMALLINT | DOUBLE | PRECISION | FLOAT | REAL**
- **DECIMAL (precision, scale)**  tipo standard SQL (aceito em ORACLE)
  - *precision* - número total de dígitos
  - *scale* - número de dígitos depois do ponto
- **NUMBER (precisão, escala)**  tipo numérico ORACLE
- **CHAR (n)** - tamanho fixo - n caracteres
- **VARCHAR (n)** - tamanho variável  ORACLE: **VARCHAR2**
  - máximo de n caracteres
- **BLOB** – *Binary Large Object*
- **CLOB** - *Character Large Object*
- **DATE | TIME | TIMESTAMP**

# SQL – Alguns tipos de dados

	<b>int10</b>	<b>int6</b>	<b>int1</b>	<b>char(n)</b>	<b>blob</b>	<b>XML</b>
Oracle 11	NUMBER(10)	NUMBER(6)	NUMBER(1)	VARCHAR2(n)	BLOB	XMLType
MS SQL Server 2005	NUMERIC(10)	NUMERIC(6)	TINYINT	VARCHAR(n)	IMAGE	XML
Sybase system 10	NUMERIC(10)	NUMERIC(6)	NUMERIC(1)	VARCHAR(n)	IMAGE	
MS Access (Jet)	Long Int or Double	Single	Byte	TEXT(n)	LONGBINARY	
TERADATA	INTEGER	DECIMAL(6)	DECIMAL(1)	VARCHAR(n)	VARBYTE(20480)	
DB2	INTEGER	DECIMAL(6)	DECIMAL(1)	VARCHAR(n)	VARCHAR(255)	
RDB	INTEGER	DECIMAL(6)	DECIMAL(1)	VARCHAR(n)	LONG VARCHAR	
INFORMIX	INTEGER	DECIMAL(6)	DECIMAL(1)	VARCHAR(n)	BYTE	
RedBrick	integer	int	int	char(n)	char(1024)	
INGRES	INTEGER	INTEGER	INTEGER	VARCHAR(n)	VARCHAR(1500)	

## CREATE TABLE

```
CREATE TABLE tabela (  
    atrib1 tipo [(tamanho)] [NOT NULL | DEFAULT valor]  
        [CHECK (condição)],  
    atrib2 tipo [(tamanho)] [NOT NULL | DEFAULT valor]  
        [CHECK (condição)],  
    ...  
    [CONSTRAINT nome da restrição]  
    PRIMARY KEY (<atributos chave primária>),  
    [CONSTRAINT nome da restrição]  
    UNIQUE (< atributos chave candidata>),  
    [CONSTRAINT nome da restrição]  
    FOREIGN KEY (<atributos chave estrangeira>)  
    REFERENCES tabelaRef [(<chave primária>)]  
        [ON DELETE CASCADE | SET NULL | SET DEFAULT]  
        [ON UPDATE CASCADE | SET NULL | SET DEFAULT],  
    [CONSTRAINT nome da restrição]  
    CHECK (condição)  
);
```



ORACLE não  
implementa  
**ON UPDATE**



ORACLE não implementa  
**SET DEFAULT**

Veremos como fazer ON UPDATE e SET  
DEFAULT na aula de Triggers mais a frente.

```
CREATE TABLE ALUNO (  
    NOME VARCHAR2(30) NOT NULL,  
    NUSP NUMBER NOT NULL,  
    IDADE NUMBER(3),  
    DATANASC DATE,  
    PRIMARY KEY (NUSP),  
    UNIQUE(NOME)  
);
```



Aluno = {Nome, Nusp, Idade, DataNasc}

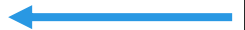
Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

```
CREATE TABLE PROFESSOR (  
    NFUNC NUMBER NOT NULL PRIMARY KEY,  
    NOME VARCHAR2(30) NOT NULL UNIQUE,  
    IDADE NUMBER(3),  
    TITULACAO CHAR(10) NOT NULL,  
    CHECK (TITULACAO IN ('MESTRE', 'DOUTOR', 'TITULAR'))  
);
```



```
CREATE TABLE ALUNO (  
    NOME VARCHAR2(30) NOT NULL,  
    NUSP NUMBER NOT NULL,  
    IDADE NUMBER(3),  
    DATANASC DATE,  
    PRIMARY KEY (NUSP),  
    UNIQUE(NOME)  
);
```

### **O Oracle irá executar:**

```
CREATE TABLE USER."ALUNO"  
    (  
        "NOME" VARCHAR2(30 BYTE) NOT NULL ENABLE,  
        "NUSP" NUMBER NOT NULL ENABLE,  
        "IDADE" NUMBER(3,0),  
        "DATANASC" DATE,  
  
        PRIMARY KEY ("NUSP")  
        USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255  
        TABLESPACE "USERS" ENABLE,  
  
        UNIQUE ("NOME")  
        USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255  
        TABLESPACE "USERS" ENABLE  
    )  
SEGMENT CREATION DEFERRED  
PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255  
NOCOMPRESS LOGGING  
TABLESPACE "USERS" ;
```

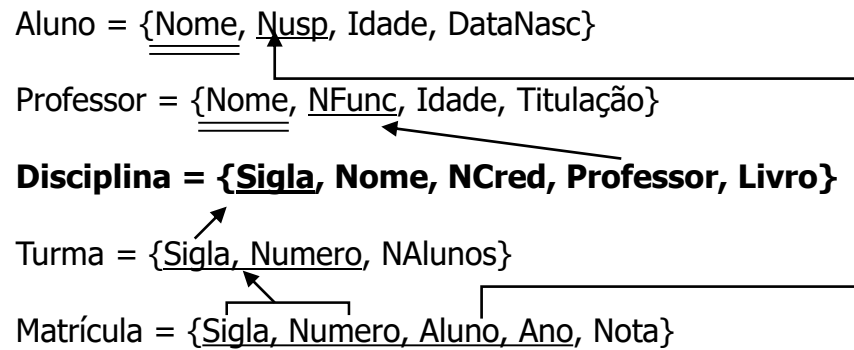
Aluno = {Nome, Nusp, Idade, DataNasc}

Professor = {Nome, NFunc, Idade, Titulação}

**Disciplina = {Sigla, Nome, NCred, Professor, Livro}**

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}



```
CREATE TABLE DISCIPLINA (  
  SIGLA CHAR(6) NOT NULL,  
  NOME VARCHAR2(30) NOT NULL,  
  NCRED NUMBER NOT NULL,  
  PROFESSOR NUMBER ,  
  LIVRO VARCHAR2(30),  
  CONSTRAINT PK_DISCIPLINA PRIMARY KEY (SIGLA),  
  CONSTRAINT FK_DISCIPLINA FOREIGN KEY (PROFESSOR)  
    REFERENCES PROFESSOR(NFUNC)  
    ON DELETE SET NULL,  
  CONSTRAINT NCREDITOS CHECK (NCRED > 0)  
);
```



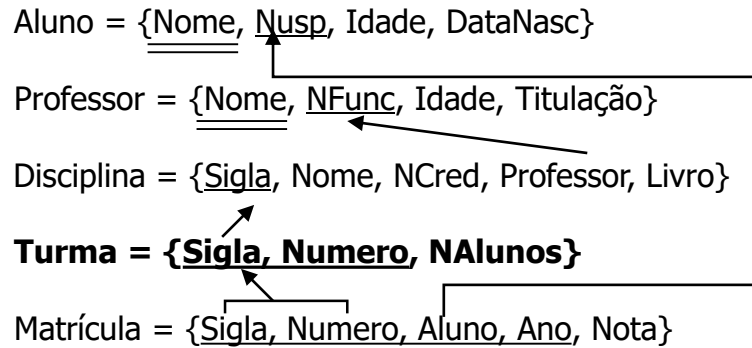
Aluno = {Nome, Nusp, Idade, DataNasc}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

**Turma = {Sigla, Numero, NAlunos}**

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}



```
CREATE TABLE TURMA (  
    SIGLA CHAR(6) NOT NULL,  
    NUMERO NUMBER NOT NULL,  
    NALUNOS NUMBER NOT NULL  
    CHECK(NAlunos <= 70),  
    PRIMARY KEY (SIGLA, NUMERO),  
    FOREIGN KEY (SIGLA) REFERENCES DISCIPLINA(SIGLA)  
    ON DELETE CASCADE  
);
```

Aluno = {Nome, Nusp, Idade, DataNasc}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}



```
CREATE TABLE MATRICULA (  
  SIGLA CHAR(6) NOT NULL,  
  NUMERO NUMBER NOT NULL,  
  ALUNO NUMBER NOT NULL,  
  ANO NUMBER(4) NOT NULL,  
  NOTA FLOAT,  
  PRIMARY KEY (SIGLA, NUMERO, ALUNO, ANO),  
  FOREIGN KEY (SIGLA, NUMERO)  
    REFERENCES TURMA(SIGLA, NUMERO)  
    ON DELETE CASCADE,  
  FOREIGN KEY (ALUNO) REFERENCES ALUNO(NUSP)  
    ON DELETE CASCADE  
);
```

# Comandos DDL

---

**ALTER TABLE** – incluir/alterar/remover definições de colunas e restrições

**ALTER TABLE *tabela* <ação>;**

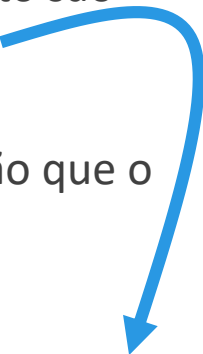
■ **<ação>:**

- **ADD *novoAtrib* *tipo* [<restrições de coluna>]**
- **ADD [CONSTRAINT *nome*] <restrição de tabela>**
- **DROP COLUMN *atributo* [CASCADE | RESTRICT]**
- **DROP CONSTRAINT *nome***
- **ALTER *atributo* DROP DEFAULT;**
- **ALTER *atributo* SET DEFAULT <valor>;**

# ALTER TABLE

---

- **ADD** *novoAtrib tipo* [*<restrições de coluna>*]
- **DROP COLUMN** *atributo* [**CASCADE** | **RESTRICT**]
  - **CASCADE** – todas as visões e restrições (*constraints*) que referenciam o atributo são removidas automaticamente
  - **RESTRICT** – atributo só é removido se não houver nenhuma visão ou restrição que o referencie



Remoção apenas das  
restrições, não dos dados.

Aluno = {Nome, Nusp, Idade, DataNasc, **CidadeOrigem**}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

ORACLE:

- **DROP COLUMN**
- **CASCADE CONSTRAINTS**
- não aceita a keyword **RESTRICT**, pois já é o *default*

**alter table Aluno add CidadeOrigem varchar(30)  
default 'São Carlos';**

**alter table Turma drop COLUMN Numero restrict;**

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

**alter table Turma drop COLUMN Numero cascade;**

Turma = {Sigla, NAlunos}

Matrícula = {Sigla, **Numero**, Aluno, Ano, Nota}

Aluno = {Nome, Nusp, Idade, DataNasc, CidadeOrigem }

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

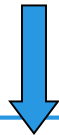
Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

**alter table** Matricula **add constraint** nota **check** (nota > 0);

**alter table** Disciplina **drop constraint** ncreditos;

**alter table** Aluno **alter** CidadeOrigem **set default** 'Sanca';



ORACLE:

```
- alter table Aluno modify (CidadeOrigem default 'Sanca');
```

Adicionar not null:

```
- alter table Aluno modify (CidadeOrigem not null);
```

Retirar not null:

```
- alter table Aluno modify (CidadeOrigem null);
```

Aluno = {Nome, Nusp, Idade, DataNasc, CidadeOrigem }

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Numero, NAlunos}

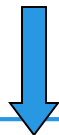
Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

O Oracle também aceita renomear atributos, por exemplo:

```
alter table Disciplina  
    ALTER TABLE Disciplina  
    RENAME COLUMN Nome TO TituloDiscip;
```

```
alter table Disciplina drop constraint ncreditos;
```

```
alter table Aluno alter CidadeOrigem set default 'Sanca';
```



ORACLE:

```
- alter table Aluno modify (CidadeOrigem default 'Sanca');
```

# Constraint - estados

---

- Uma constraint pode estar em 4 estados:
  - Enable , Validade: ativa e checando valores anteriores à sua criação
  - Enable , Novalidade: ativa, mas não checando valores anteriores à sua criação
  - Disable , Validade: desativada para valores novos, mas ainda checando updates em valores existentes
  - Disable , Novalidade: desativada totalmente (equivalente a DISABLE apenas)
- Comandos:

```
ALTER TABLE table [ENABLE | DISABLE] CONSTRAINT constraint_name;
```

```
ALTER TABLE table [ENABLE | DISABLE] CONSTRAINT constraint_name;
```

- No caso de VALIDATE/NOVALIDATE, o controle só ocorre no momento da criação da constraint via ALTER TABLE

```
ALTER TABLE table ADD CONSTRAINT constraint_name constraint_type (columns) [ENABLE | DISABLE]  
[VALIDATE | NOVALIDATE];
```




# Comandos DDL

---

**DROP TABLE** - exclui uma tabela da base de dados

**DROP TABLE** *tabela* [**CASCADE** | **RESTRICT**];

- **CASCADE**: todas as visões e restrições que referenciam a tabela são removidas automaticamente
- **RESTRICT**: a tabela é removida somente se não for referenciada em nenhuma restrição ou visão



ORACLE remove apenas as restrições, não remove os dados das tabelas que fazem a referência.

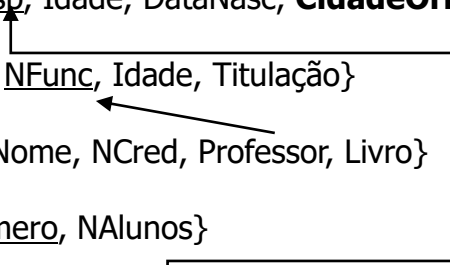
Aluno = {Nome, Nusp, Idade, DataNasc, **CidadeOrigem**}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Numero, NAlunos}

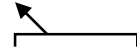
Matrícula = {Sigla, Numero, Aluno, Ano, Nota}



**drop table Turma restrict;**

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}



**drop table Turma cascade;**

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

**- CASCADE apenas das CONSTRAINTS**

# Comandos DDL

---

Como *dropar* todas as tabelas do esquema:

1) Executar o seguinte comando como statement:

```
select 'drop table ' || table_name || ' cascade  
constraints;' from user_tables
```

2) Copiar a saída, colar na planilha SQL, e rodar como script.

# Seqüência (*Sequence*)

---

Gerador de números sequenciais que retornam um valor diferente a cada acesso;

É usualmente utilizada para gerar chaves primárias.

```
CREATE SEQUENCE nome_sequencia  
[START WITH valor_inicial] /*(default 1)*/  
[INCREMENT BY incremento] /*(default 1)*/  
[MAXVALUE valor_maximo / NOMAXVALUE]  
[MINVALUE valor_minimo / NOMINVALUE]  
[CYCLE / NOCYCLE]  
[CACHE/NOCACHE]
```

# Seqüência (*Sequence*)

---

Gerada  
diferente  
É usada

Usando uma seqüência:

- `SELECT <sequence name>.NextVal  
FROM dual`

Metadados das seqüências:

- `DESC SYS.USER_SEQUENCES`
- `SELECT *  
FROM SYS.USER_SEQUENCES`

`[MINVALUE value | MAXVALUE | NO MINVALUE]`  
`[CYCLE / NOCYCLE]`

# Seqüência (*Sequence*)

---

A sequencia pode ser usada de duas maneiras:

## 1) **INSERT**

```
INSERT INTO TABLE tabela VALUES(nomeSeq.nextval)
```

## 2) **CREATE** (semelhante ao tipo SERIAL do PostgreSQL)

```
CREATE TABLE teste(  
    ID INTEGER DEFAULT nomeSeq.nextval  
)
```

# Sinônimo (*Synonym*)

---

- *Alias* para um objeto de esquema
- Pode ser público – acessível por todos os usuários – ou privado – pertencentes a um determinado esquema.
- Utilidade:
  - segurança
  - transparência
  - simplicidade
  - ...

```
CREATE [PUBLIC] SYNONYM sinonimo FOR objeto
```

# Sinônimo (*Synonym*)

---

- *Alia* Exemplo:
- Pod
- priv
- Util `CREATE SYNONYM t`  
`FOR tabela_com_nome_comprido`
  - 
  - transparência
  - simplicidade
  - ...

```
CREATE [PUBLIC] SYNONYM sinonimo FOR objeto
```



# DDL (INSERT)

---

# Comandos DML

---

**INSERT** – insere uma ou mais tuplas em uma tabela

-Inserção de 1 tupla:

```
INSERT INTO tabela [(atrib1,atrib2,...)]  
VALUES (valor1, valor2,...)
```

-Inserção de múltiplas tuplas:

```
INSERT INTO tabela [(atrib1,atrib2,...)]  
<comando SELECT>
```

# Exercício

Aluno = {Nome, Nusp, Idade, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota, Frequencia}

Inserir os seguintes dados:

- aluna de nome Juliana, nro usp 222, nascida em 10 de abril de 1989, com cidade de origem *default*
- disciplina SCC518, Banco de Dados, com 4 créditos.
- matrícula da Juliana na disciplina SCC518, turma 1

Criar uma tabela para os alunos menores de idade e alimentar com os alunos menores da tabela Aluno

# Exercício

Aluno = {Nome, Nusp, Idade, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota, Frequencia}

Inserir os seguintes dados:

- aluna de nome Juliana, nro usp 222, nascida em 10 de abril de 1989, com cidade de origem *default*

```
INSERT INTO Aluno (NUSP, Nome, Idade, DataNasc) VALUES (222, 'Juliana', 20, '10/04/1989');
```

- disciplina SCC518, Banco de Dados, com 4 créditos

```
INSERT INTO Disciplina VALUES ('SC518', 'Banco de Dados', 4, 10, 'Fundamentos de Bancos de Dados');
```

- matrícula da Juliana na disciplina SCC518, turma 1

```
INSERT INTO Turma VALUES ('SC518', 1, 1);
```

**COMMIT;**

```
INSERT INTO Matricula (Sigla, Numero, Aluno, Ano) VALUES ('SC518', 1, 222, 2010);
```

- Criar uma tabela para os alunos menores de idade e alimentar com os alunos menores da tabela Aluno

# Exercício

Aluno = {Nome, Nusp, Idade, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota, Frequencia}

Criar uma tabela para os alunos menores de idade e alimentar com os alunos menores da tabela Aluno

```
CREATE TABLE Aluno_Menor(  
  NUSP NUMERIC(7) NOT NULL,  
  Nome VARCHAR(100) NOT NULL,  
  Idade SMALLINT,  
  DataNasc DATE,  
  CidadeOrigem VARCHAR(100) DEFAULT 'Sao Carlos',  
  
  CONSTRAINT aluno_menor_pk PRIMARY KEY(NUSP),  
  CONSTRAINT aluno_menor_un UNIQUE(Nome),  
  CONSTRAINT aluno_menor_ck CHECK(Idade < 15)  
);  
  
INSERT INTO aluno_menor SELECT * FROM aluno WHERE Idade < 18;
```

# Comandos DML

---

**UPDATE** – modifica o valor de um atributo em uma ou mais tuplas da tabela

```
UPDATE tabela SET  
    atributo1 = <valor ou expressão>,  
    atributo2 = <valor ou expressão>,  
    ...  
WHERE <condição de localização>
```

# Comandos DML

---

**DELETE** – remove uma ou mais tuplas da tabela

```
DELETE FROM tabela1 [FROM tabela2]  
[WHERE <condição de localização>]
```

# Exercícios

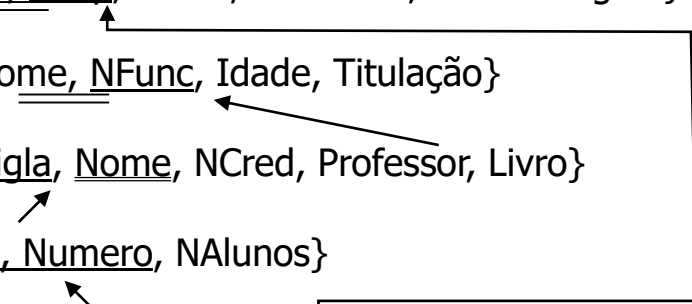
Aluno = {Nome, Nusp, Idade, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Numero, NAlunos}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota, Frequencia}



Atualizar os seguintes dados:

- alterar para 70% a frequência de todos os alunos com nota acima de 5.0 e frequência abaixo 70%

```
UPDATE MATRICULA SET FrequenciaPorc = 70
```

```
WHERE Nota >= 5 AND FrequenciaPorc < 70;
```

- acrescentar um crédito para as disciplinas do departamento de Matemática(SM)

```
UPDATE Disciplina SET NCred = NCred+1
```

```
WHERE Sigla LIKE 'SM%';
```

Remover os seguintes dados

- matrícula dos alunos da turma 1 de SC241

```
DELETE FROM MATRICULA WHERE Sigla = 'SC241' AND Numero = 1;
```

- disciplinas com número de créditos superior a 6

```
DELETE FROM MATRICULA WHERE Ncred > 6;
```



---

# Prática 2