



Serviço Nacional de Aprendizagem Industrial

PELO FUTURO DO TRABALHO

SQL

Structured Query Language

Prof. Dr. Halley Wesley Gondim
halley.was@gmail.com

SQL - Tirinha



SQL - Histórico

- ✓ Versão original desenvolvida pela IBM (Lab. De Pesquisa San José)
- ✓ Originalmente chamada de Sequel ("Structured English Query Language" (Linguagem de Consulta Estruturada em Inglês))
- ✓ SQL (Structured Query Language – Linguagem de Consulta Estruturada)
- ✓ 1986 – American National Standards Institute (ANSI) e a International Standards Organization (ISO) publicaram padrões para SQL. (SQL-86)

SQL - Histórico

- ✓ SQL (ISO/IEC 9075-x) foi revisto em:
 - SQL-99
 - SQL-2003
 - SQL-2008
 - SQL-2016
- ✓ Linguagem de Definição de Dados (DDL)
 - Comandos definição de esquemas, exclusão, criação de índices e modificação nos esquemas de relações

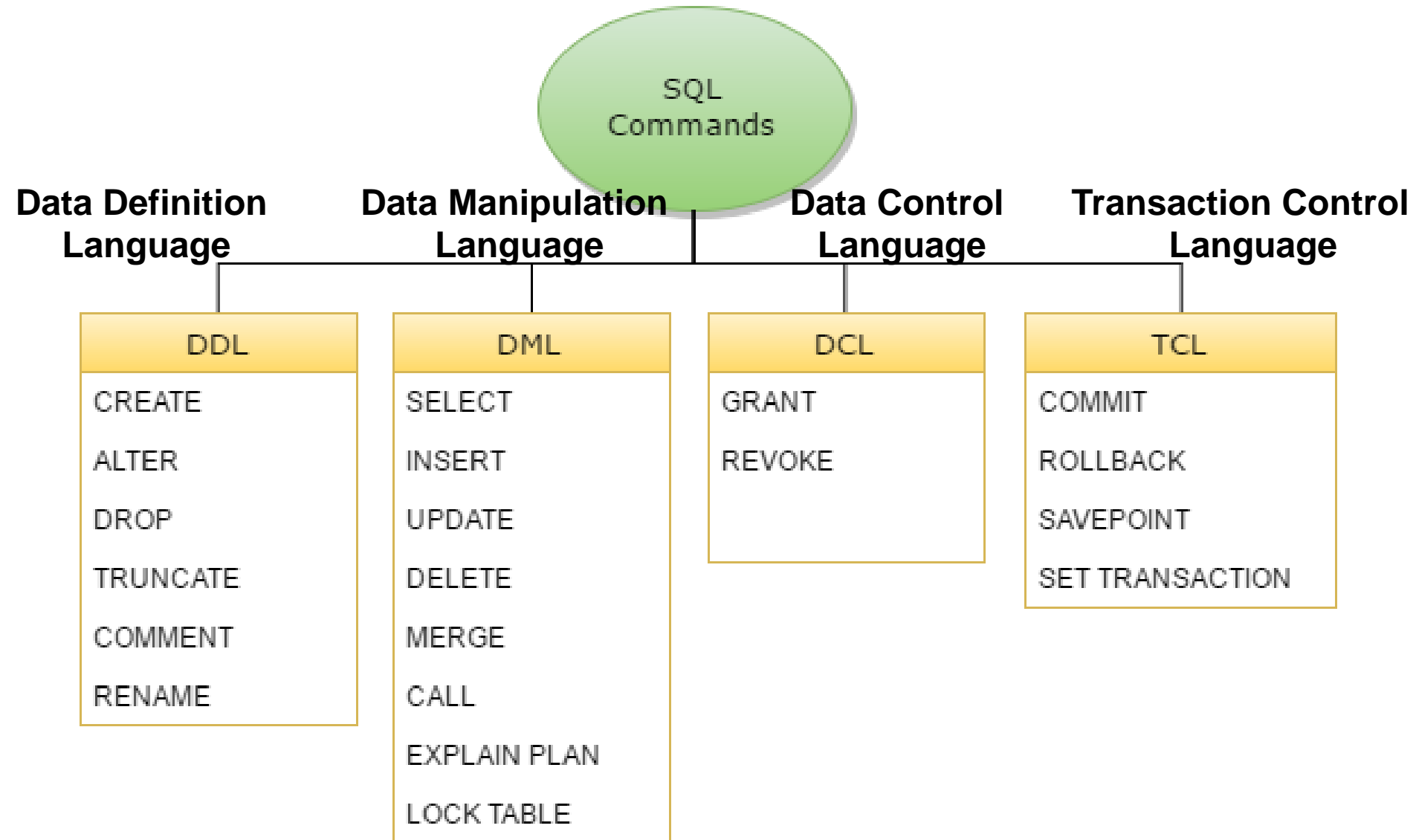
SQL - Histórico

- ✓ Linguagem Interativa de Manipulação de Dados (DML)
 - Linguagem de consulta, inserção, exclusão e modificação
- ✓ Definição de visões
- ✓ Autorização
- ✓ Integridade
- ✓ Controle de transações
 - Inclui comandos para especificar iniciação e finalização de transações.

SQL – Estruturas Básicas

- ✓ SQL permite uso de valores nulos
 - (desconhecidos / inexistentes)
- ✓ A estrutura básica de expressão em SQL consiste:
 - **Select** (Projeção de álgebra relacional – relacionar atributos desejados)
 - **From** (Produto cartesiano, associação entre relações pesquisadas)
 - **Where** (Seleção do predicado - condições)

SQL – Categorias



SQL – DDL

✓ Uma **DDL (Linguagem de Definição de Dados)** permite ao usuário definir novas tabelas e elementos associados.

Obs: A maioria dos bancos de dados de SQL comerciais tem extensões proprietárias no DDL.

✓ SQL DDL permite não só **especificação** de um conjunto de relações, como também **informações** acerca de cada uma **das relações**

SQL – DDL

- ✓ O **esquema** de cada relação (tabela).
- ✓ O **domínio dos valores** associados a cada atributo
- ✓ **Regras de integridade**
- ✓ O conjunto de **índices** para manutenção de cada relação
- ✓ Informações sobre **segurança** e autoridade sobre cada relação
- ✓ A estrutura de **armazenamento** físico de cada relação no disco.

SQL – Definição de Esquema em SQL

Base de Dados – Criar nossos bancos.

```
1  /*CRIAR NOVO DATABASE*/
2  CREATE DATABASE nome;
3
4  /*ALTERAR O NOME DO DATABASE*/
5  ALTER DATABASE nome RENAME TO novo_nome
6
7  /*ALTERAR O PROPRIETÁRIO DO DATABASE*/
8  ALTER DATABASE nome OWNER TO novo_dono
9
10 /*APAGAR DATABASE*/
11 DROP DATABASE novo_nome;
12
```

SQL – Tipos de domínio em SQL

- ✓ **Char(n)** – cadeia de caractere – tamanho fixo
 - ✓ **Varchar(n)** – cadeia caractere - variável – ($\leq n$)
 - ✓ **Integer** – inteiro
 - ✓ **Numeric** - ponto flutuante, precisão em cálculos
 - ✓ **Serial / BigSerial** – inteiro com incremento automático
 - ✓ **Date** – Ano(4 dig.), mês e dia
 - ✓ **Time** – horas, minutos e segundos
 - ✓ **Clob** – texto “infinito”
 - ✓ **Blob** – armazenamento de até 4Gb de dados.
 - ✓ Como é booleano?
- (<https://www.postgresql.org/docs/11/index.html>)

SQL – Definição de Esquema em SQL

✓ Criando uma tabela

▪ **CREATE TABLE** r ($A_1D_1, A_2D_2, \dots, A_ND_N$

< REGRAS DE INTEGRIDADE₁> ,

...

< REGRAS DE INTEGRIDADE_k>)

R = Tabela

A = Atributos

D = Domínio

SQL – Definição de Esquema em SQL

- ✓ Regras de integridades permitidas englobam:
 - **PRIMARY KEY** ($A_{j1}, A_{j2}, \dots, A_{j1m}$)
 - **CHECK** (P)

Os atributos $A_{j1}, A_{j2}, \dots, A_{j1m}$ formam a chave primária da relação

Check especifica um predicado P que precisa ser satisfeito por todas as tuplas em uma tabela/relação

SQL – Definição de Esquema em SQL

```
1 CREATE TABLE nome_da_tabela (  
2 atributo_chave SERIAL PRIMARY KEY,  
3 atributo_1 VARCHAR(80),  
4 atributo_2 NUMERIC(7,2)  
5 )
```

```
1 CREATE TABLE nome_da_tabela (  
2 atributo_chave SERIAL PRIMARY KEY,  
3 atributo_1 VARCHAR(80),  
4 atributo_2 NUMERIC(7,2) CHECK(atributo_2 > 0),  
5 atributo_3 VARCHAR(80) CHECK (atributo_3 IN ('M','F','A')),  
6 atributo_estrangeiro INTEGER,  
7 FOREIGN KEY (atributo_estrangeiro)  
8 REFERENCES nome_tabela_estrangeira  
9 (atributo_chave_tabela_estrangeira)  
10 )  
11
```

SQL – Tipos de domínio em SQL

✓ **Serial / BigSerial** – inteiro com incremento automático

```
1  /*INCREMENTO*/
2  CREATE SEQUENCE nome_tabela_seq;
3  CREATE TABLE nome_da_tabela (
4      nome_da_coluna integer PRIMARY KEY
5      DEFAULT nextval('nome_tabela_seq') NOT NULL
6  );
7
8  /*INCREMENTO COM SERIAL*/
9  CREATE TABLE nome_da_tabela(
10 nome_da_coluna SERIAL PRIMARY KEY
11 )
```

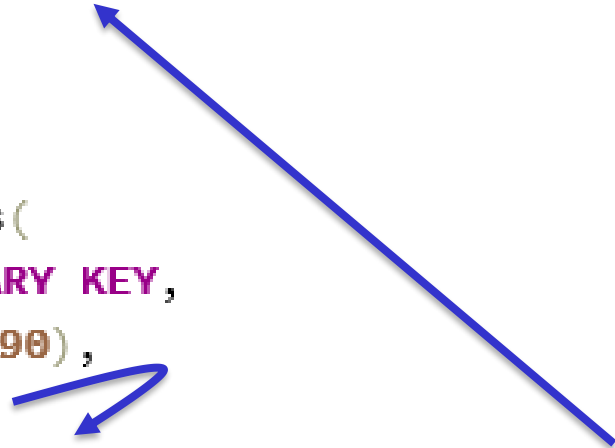
SQL – Sequence

✓Corpo de uma sequence

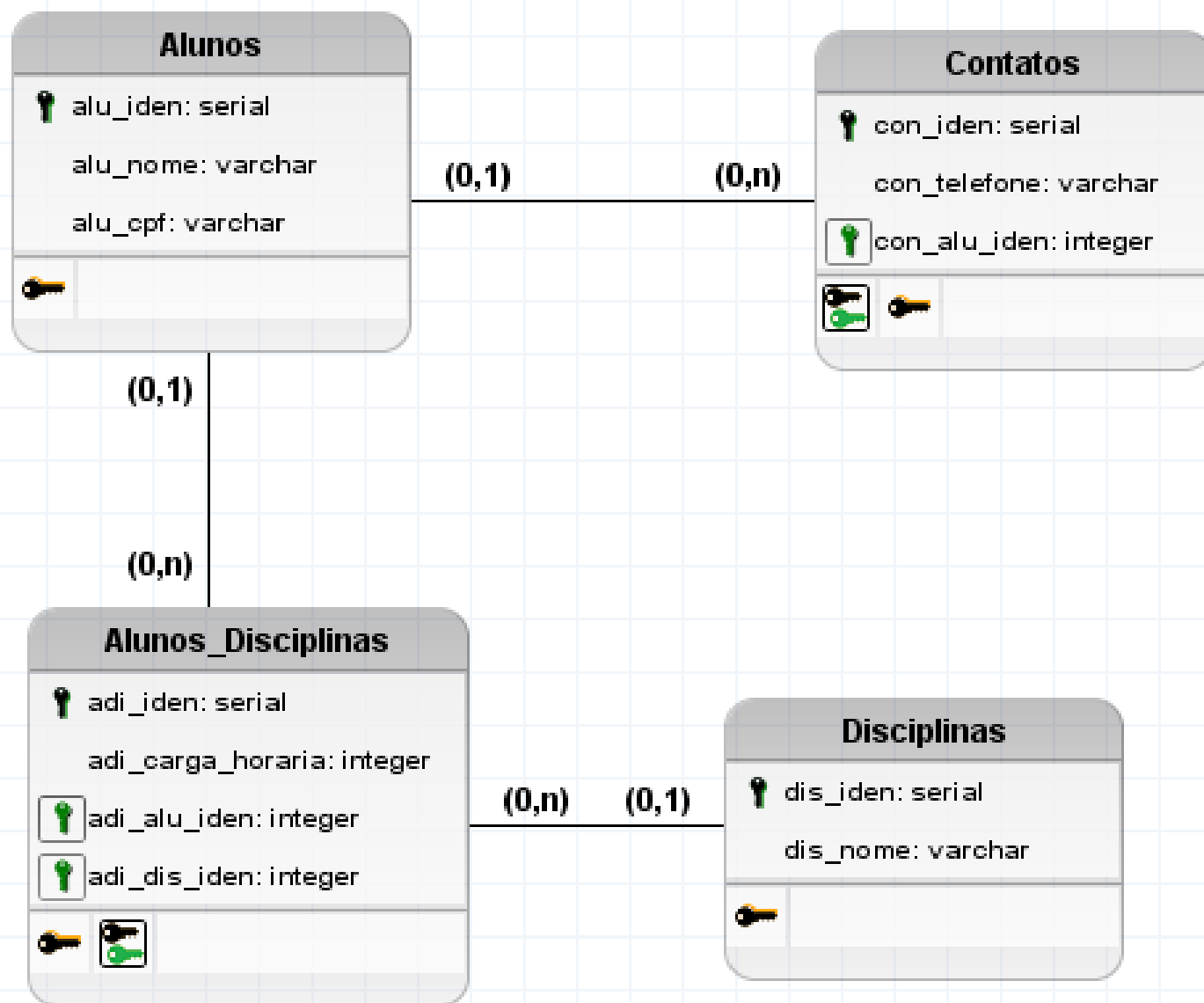
```
1 CREATE SEQUENCE public.nome_da_tabela_seq
2     INCREMENT 1
3     START 1
4     MINVALUE 1
5     MAXVALUE 2147483647
6     CACHE 1;
7
```


SQL – Create table exemplo

```
1  /*TABELA COM ATRIBUTO AUTOINCREMENTO SERIAL*/
2  CREATE TABLE alunos(
3  alu_iden serial PRIMARY KEY,
4  alu_nome varchar(80)
5  )
6
7  CREATE TABLE contatos(
8  con_iden serial PRIMARY KEY,
9  con_telefone varchar(90),
10 con_alu_iden integer,
11 FOREIGN KEY (con_alu_iden) REFERENCES alunos (alu_iden)
12 )
```



SQL – Crie as seguintes tabelas



SQL – DDL tabelas

```
1  CREATE TABLE alunos (  
2      alu_iden serial PRIMARY KEY,  
3      alu_nome varchar,  
4      alu_cpf varchar  
5  );  
6  
7  CREATE TABLE contatos (  
8      con_iden serial PRIMARY KEY,  
9      con_telefone varchar,  
10     con_alu_iden integer,  
11     FOREIGN KEY (con_alu_iden) REFERENCES alunos (alu_iden)  
12 );  
13
```

SQL – DDL tabelas

```
14 CREATE TABLE disciplinas (  
15     dis_iden serial PRIMARY KEY,  
16     dis_nome varchar  
17 );  
18  
19 CREATE TABLE alunos_disciplinas (  
20     adi_iden serial PRIMARY KEY,  
21     adi_carga_horaria integer,  
22     adi_alu_iden integer,  
23     adi_dis_iden integer,  
24     FOREIGN KEY (adi_alu_iden) REFERENCES alunos (alu_iden),  
25     FOREIGN KEY (adi_dis_iden) REFERENCES disciplinas (dis_iden)  
26 );  
27
```