INSTITUTO FEDERAL
CATARINENSE

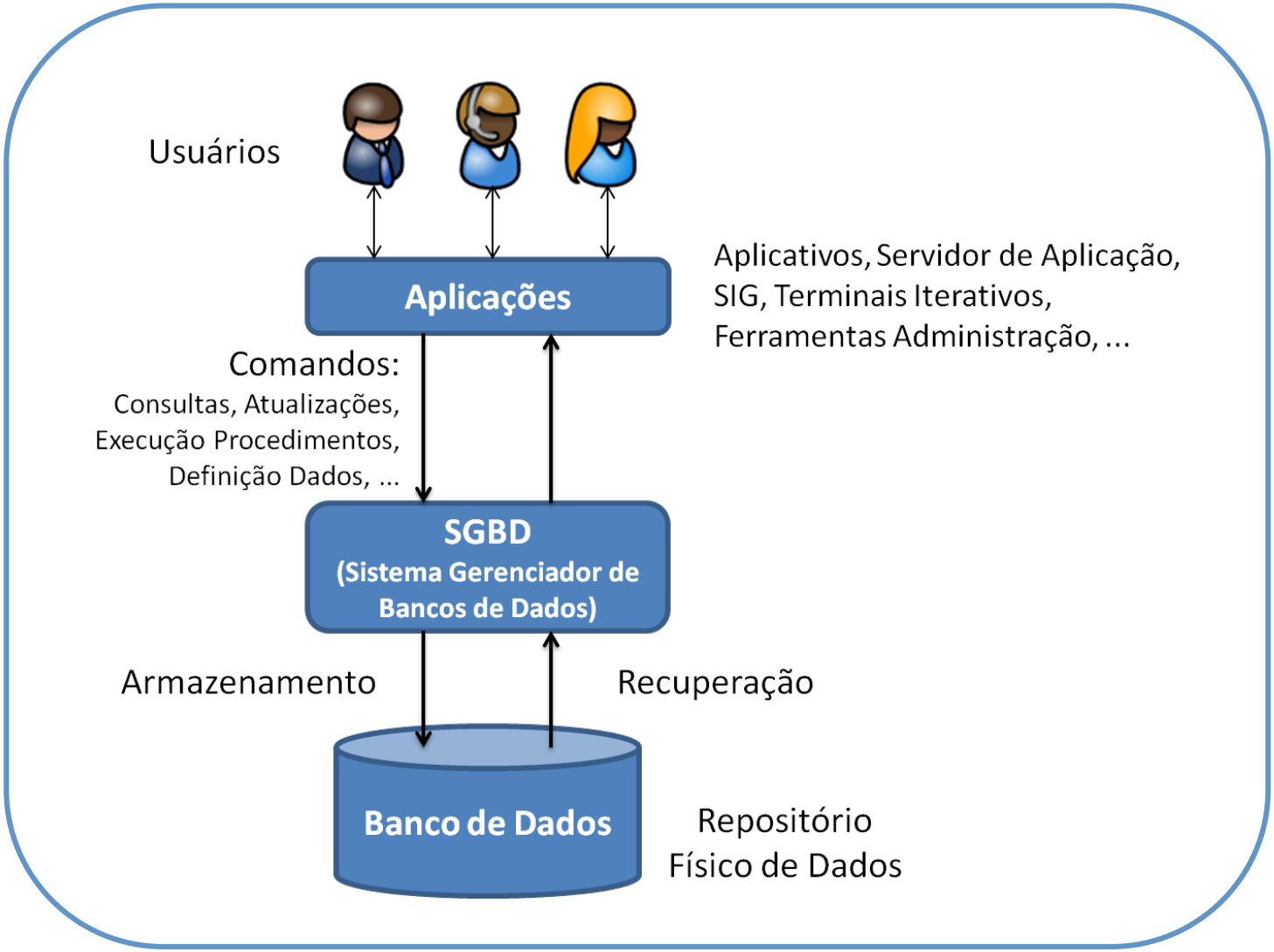
Linguagem Estruturada de Consultas (SQL)

Prof: Aldelir Fernando Luiz

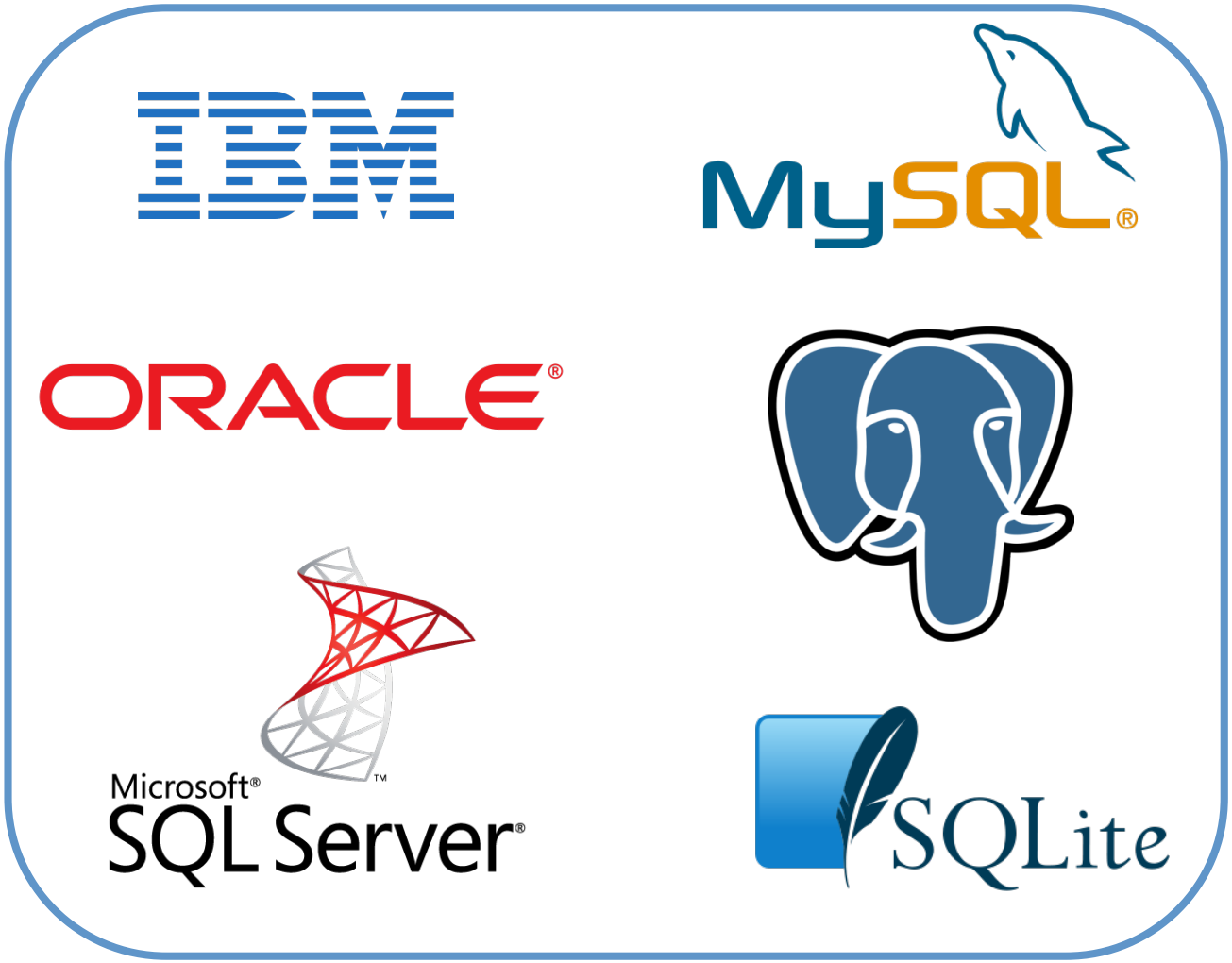
Disciplina: Banco de Dados I
Curso: Bacharelado em Ciência da Computação

IFC – Instituto Federal de Educação, Ciência e Tecnologia Catarinense,
Câmpus Blumenau

Interação num SBD



Interação num SBD



SQL

- Uma linguagem de consulta é a linguagem por meio da qual os usuários obtêm informações do banco de dados
- O nome "SQL" advém de "Structured Query Language” (ou Linguagem Estruturada de Consultas)
- Fundamentada no modelo relacional de Codd (1970), inicialmente recebeu o nome de SEQUEL ("Structured English Query Language”) (IBM, 1974)
  - Por razões jurídicas, em 1977 ela foi modificada, aprimorada e revisada, e renomada para apenas SQL

SQL

- Cronologia
  - SEQUEL (IBM, 74)
  - Implementação no SQL/DS (IBM, 81)
  - Desde 1983, padrão *de facto*
  - Primeira padronização em 1986, revisada em 1989 (SQL-89)
  - Segunda padronização em 1992 (SQL-92)
  - Terceira padronização em 1999 (SQL-3 ou SQL-99)

INSTITUTO FEDERAL
CATARINENSE

SQL

- A SQL assume um papel muito importante nos SGBDs da atualidade, tendo os seguintes enfoques:
  - Linguagem interativa de consulta
  - Linguagem de declaração para acesso a banco de dados
  - Linguagem de administração de banco de dados

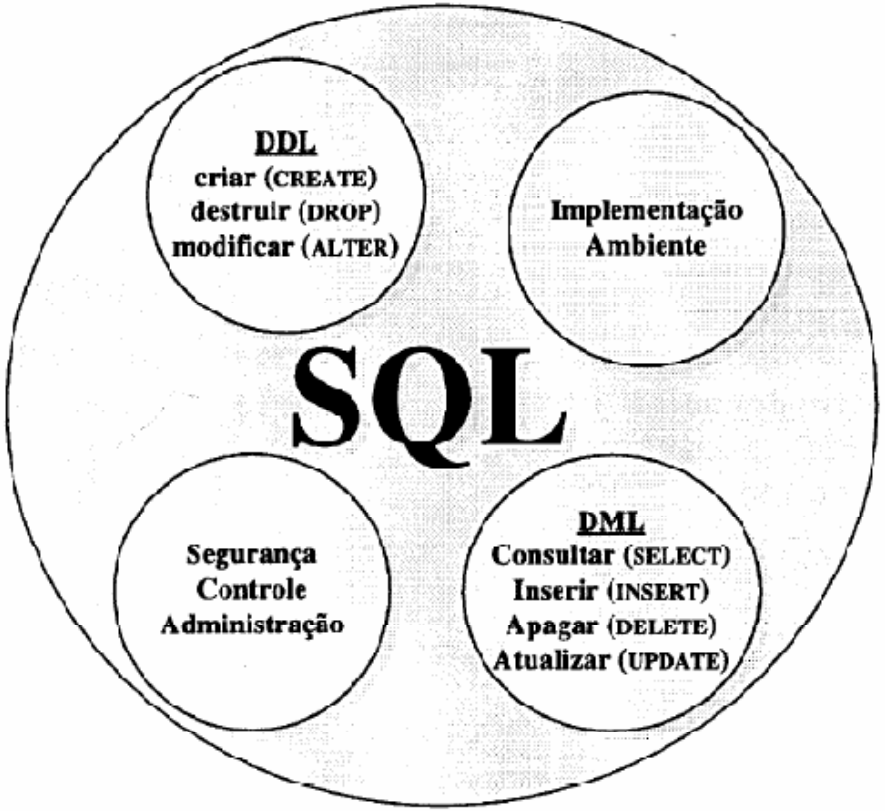
INSTITUTO FEDERAL
CATARINENSE

SQL

- Independência de fabricante
- Usa inglês estruturado de alto nível
- Permite consultas interativas
- Múltiplas visões dos dados
- Definição dinâmica de dados e também de estruturas de dados

SQL

- A SQL pode manipular objetos de diferentes classes entre as funções de um SGBD



SQL: conceitos importantes

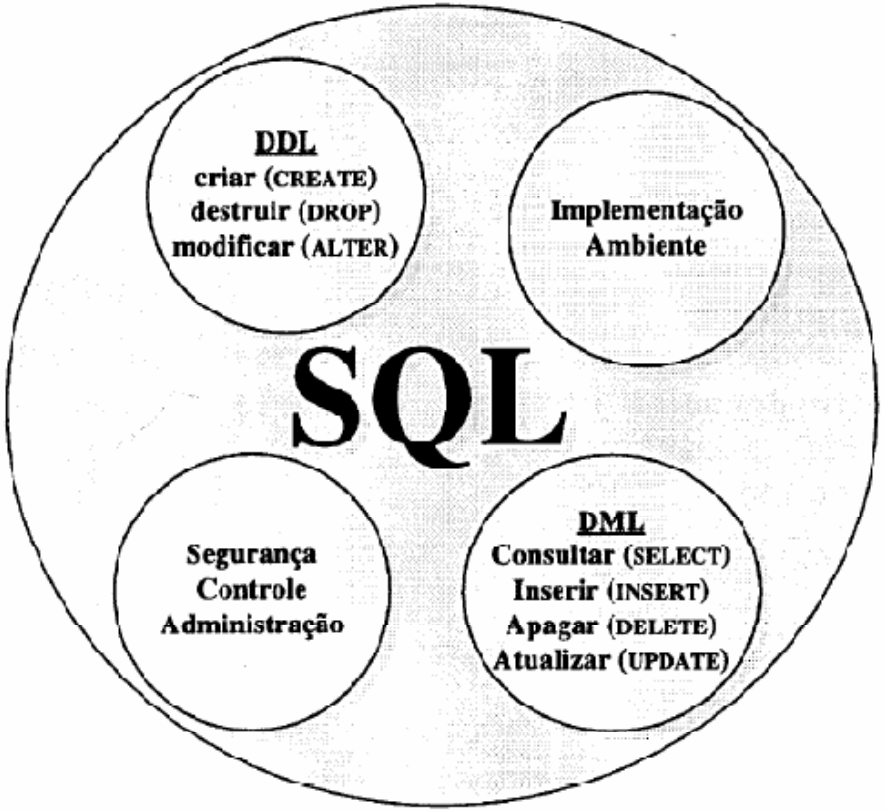
- Atributo/Coluna
  - É a menor unidade de informação existente em um arquivo de banco de dados
- Registro
  - Conjunto de campos. Entidade que identifica entrada única num banco de dados
- Tabela
  - Representa a estrutura de armazenamento de dados dos sistemas (p. ex.: conjunto de campos e registros)

SQL: conceitos importantes

- Domínio
  - especifica o conteúdo dos atributos/campos/colunas
    - Elementares (p. ex: tipos primitivos)
    - Definidos pelo usuário
- Esquema (schema)
  - É uma coleção de objetos num banco de dados; por exemplo: tabelas, índices, visões, etc.

SQL

- A SQL pode manipular objetos de diferentes classes entre as funções de um SGBD

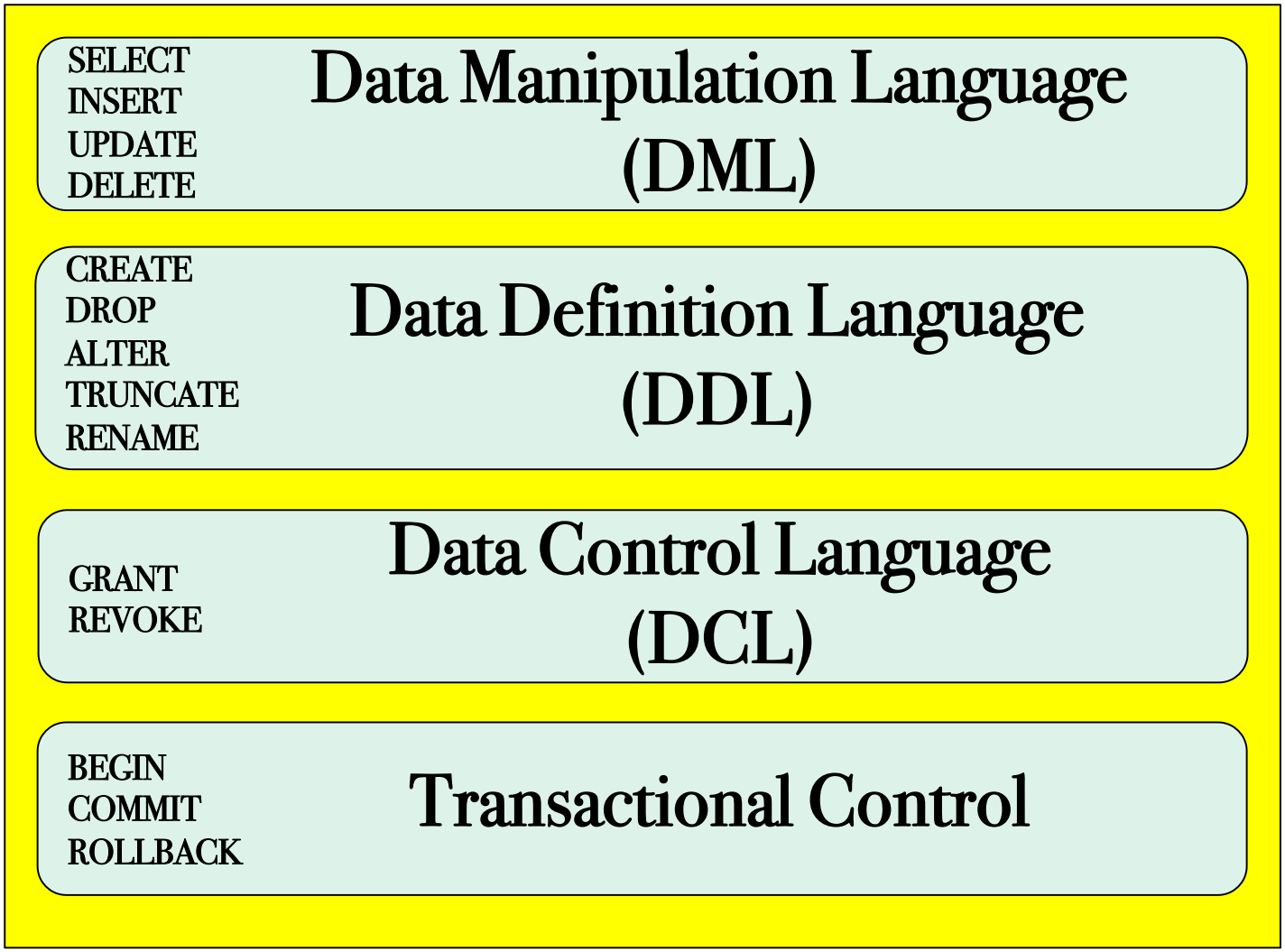


INSTITUTO FEDERAL
CATARINENSE

SQL

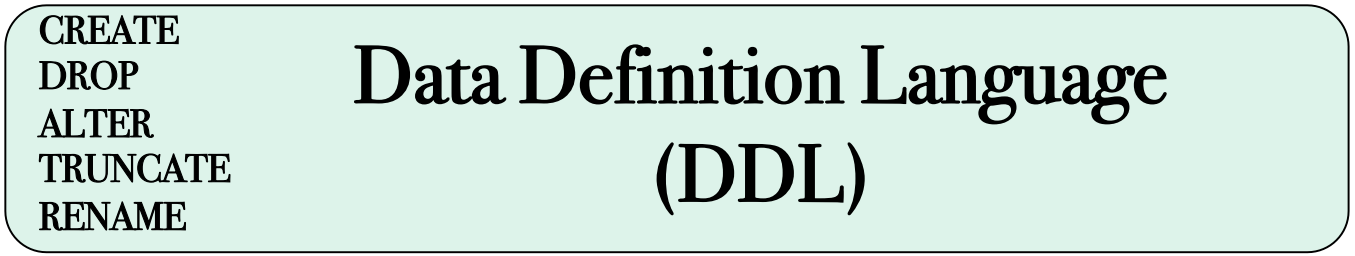
- Linguagem de Definição de Dados (DDL)
- Linguagem de Manipulação de Dados (DML)
- Linguagem de Controle de Dados (DCL)
- Controle de Transações (TCL)

SQL (Comandos)



SQL – DDL

- DDL (Data Definition Language)
  Linguagem de Definição dos dados
  - Utilizada para a criação das estruturas de dados e objetos de um banco de dados



SQL – DDL

- A partir da DDL é possível especificar
  - O esquema de cada relação
  - As relações que compõem um banco de dados
  - O domínio dos valores associados de cada atributo de uma relação
  - Restrições de integridade
  - O conjunto de índices
  - Visões de dados
  - etc.

INSTITUTO FEDERAL
CATARINENSE

SQL – DDL

- Criação de um BD ou schema
  - A especificação SQL ANSI não oferece suporte para tal comando
    - BDs/Schemas são criados via ferramentas do SGBD
  - Alguns SGBDs admitem um comando para este fim

CREATE DATABASE

CREATE SCHEMA

SQL – DDL

- **CREATE DATABASE / SCHEMA**
  - Cria um novo banco de dados
  - Sintaxe pode mudar a depender do SGBD

```
CREATE DATABASE <nome_do_BD> [argumentos]
```

```
CREATE SCHEMA <nome_do_schema> [argumentos]
```

- **DROP DATABASE / SCHEMA**
  - Exclui um banco de dados e todos os objetos nele existentes
  - Sintaxe pode mudar a depender do SGBD

```
DROP DATABASE <nome_do_BD>
```

```
DROP SCHEMA <nome_do_schema>
```

SQL – DDL

- **CREATE DATABASE / SCHEMA**
  - Em relação ao nome a ser dado ao banco de dados, é importante frisar que nem todo nome é possível (alguns exemplos a seguir)

Válido	Inválido
nome_banco	nome banco
3nome_banco	nome-banco
banco_p**á**i	nome**%**banco
Nome**B**anco	

  - Embora acentos e caracteres especiais sejam aceitos no nome dos bancos de dados, é uma boa prática evitá-los!

SQL – DDL

- **`CREATE TABLE`**

SQL – DDL

- **CREATE TABLE**
  - Após a criação de um banco de dados, procede-se com a criação das tabelas para atender a demanda específica da aplicação
  - Não é possível criar mais de uma tabela com mesmo nome em um banco de dados
  - Para a criação de tabela num banco de dados no MySQL, utiliza-se a palavra-chave **CREATE TABLE**

```
CREATE TABLE nome_tabela
(
   <nome_da_coluna1> <tipo_da_coluna1>
[<atributos_da_coluna1>],
   ...
   <nome_da_coluna> <tipo_da_coluna> [<atributos_da_coluna>]

);
```

SQL – DDL

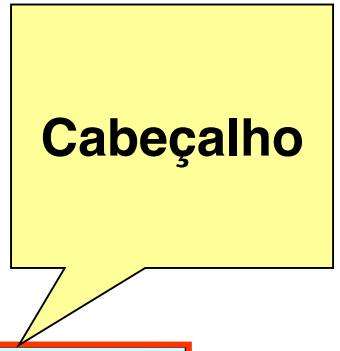
- **CREATE TABLE**
  - Tabelas consistem em objetos de banco de dados que contêm todos os dados existentes num banco de dados
  - Nestas, os dados são organizados de maneira lógica em um formato de linha-e-coluna semelhante ao de uma planilha excel
  - Cada linha representa um registro exclusivo e cada coluna representa um campo no registro

SQL – DDL

- **CREATE TABLE**
  - Tabelas consistem em objetos de banco de dados que contém todos os dados existentes num banco de dados
  - Nestas, os dados são organizados de maneira lógica em um formato de linha-e-coluna semelhante ao de uma planilha excel
  - Cada linha representa um registro exclusivo e cada coluna representa um campo no registro
  - Uma tabela é conjunto não ordenado de linhas, onde cada linha é composta por uma série de campos
  - Cada atributo/coluna é identificado por um nome de coluna

SQL – DDL

- **Exemplo de tabela**



Aluno

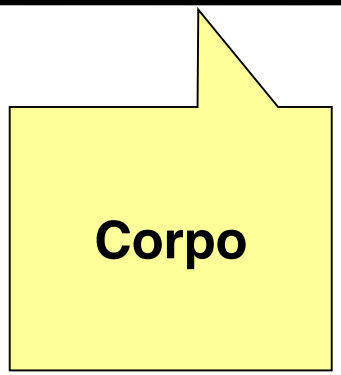
Nome	CPF	Endereço	DataNasc
Renata	01035	Rua das Flores, 210	12/11/1980
Vânia	02467	Capote Valente, 35	03/07/1976
Maria	01427	São Diego 310/34	20/02/1985

SQL – DDL

- **Exemplo de tabela**

Aluno

Nome	CPF	Endereço	DataNasc
Renata	01035	Rua das Flores, 210	12/11/1980
Vânia	02467	Capote Valente, 35	03/07/1976
Maria	01427	São Diego 310/34	20/02/1985



SQL – DDL

- **Exemplo de tabela**
  - Exemplo de chave primária numa tabela

Aluno

Nome	CPF	Endereço	DataNasc
Renata	01035	Rua das Flores, 210	12/11/1980
Vânia	02467	Capote Valente, 35	03/07/1976
Maria	01427	São Diego 310/34	20/02/1985

SQL – DDL

- **Exemplo de tabela**
  - Exemplo de chave estrangeira numa tabela

Nome	Matrícula	CPF	Curso
Renata	01035	701034263890	1
Vânia	02467	693529876987	2
Maria	01427	347685784432	1

SQL – DDL

- **Exemplo de tabela**
  - Exemplo de referência de chave estrangeira num banco de dados

Nome	Matrícula	CPF	Curso
Renata	01035	701034263890	1
Vânia	02467	693529876987	2
Maria	01427	347685784432	1

Codigo	Descrição
1	Ciência da Computação
2	Administração de Empresas
3	Ciências Jurídicas e Sociais

SQL – DDL

- Comandos para definição de objetos num esquema ou banco de dados
  - **`CREATE TABLE`**
    - define a estrutura da tabela, suas restrições de integridade e cria uma tabela vazia
  - **`ALTER TABLE`**
    - modifica a definição de uma tabela
    - define restrições de integridade (RIs)
  - **`DROP TABLE`**
    - remove uma tabela com todas as suas tuplas/ocorrências/registros

SQL – DDL

CREATE TABLE

- Exemplo:

```
CREATE TABLE CURSO (
 COD_CURSO INT NOT NULL,
 DEN_CURSO VARCHAR(50) NOT NULL,
 NUM_PERIODOS_CURSO INT,
 IND_TURNO_CURSO ENUM('M', 'V', 'N', 'I'),
 VAL_CH_CURSO DECIMAL(5,2),
 NUM_MAX_INTEGRALIZACAO INT,
 COD_DEPTO INT NOT NULL,
 PRIMARY KEY (COD_CURSO)
);
```

SQL – DDL

CREATE TABLE

- continuação…

```
CREATE TABLE ALUNO (
 COD_MAT_ALUNO BIGINT NOT NULL PRIMARY KEY,
 NOM_ALUNO VARCHAR(60) NOT NULL,
 DES_END_ALUNO VARCHAR(250),
 NUM_CPF_ALUNO BIGINT NOT NULL,
 COD_CURSO_ALUNO INT UNIQUE NOT NULL,
 IND_SEXO_ALUNO ENUM('M', 'F'),
FOREIGN KEY (COD_CURSO_ALUNO)
 REFERENCES CURSO (COD_CURSO)
);

DROP TABLE CURSO;
DROP TABLE ALUNO;
```

SQL – DDL

COD_CURSO	DEN_CURSO	NUM_PERIODOS_CURSO	IND_TURNO_CURSO	VAL_CH_CURSO	NUM_MAX_INTEGRALIZACAO	COD_DEPTO

COD_MAT_ALUNO	NOM_ALUNO	DES_END_ALUNO	NUM_CPF_ALUNO	COD_CURSO	IND_SEXO_ALUNO

SQL – DDL

- Note que a criação da tabela requer a especificação dos tipos de dados para as colunas que irão compô-la
- O MySQL permite o armazenamento em diversos tipos de dados
- Por isso, é importante conhece-los para que seja possível criar as tabelas de maneira correta e otimizada, de acordo com os dados que ela armazenará

SQL – DDL

- Os tipos podem ser agrupados da seguinte forma
  - **Numéricos**
  - **Alphanuméricos (ou textuais)**
  - **Temporais**
  - **Binários**

SQL – DDL

- **Tipos Numéricos**
  - **TINYINT** (1) de –128 até 127
  - **SMALLINT** (2) de –32768 até 32767
  - **MEDIUMINT** (3) de –8388608 até 8388607
  - **INT** (4) de –2147483648 até 2147483647
  - **BIGINT** (8) de –9223372036854775808 até 9223372036854775807

  - **FLOAT**
  - **DOUBLE**
  - **DECIMAL**(P, S)

SQL – DDL

- **Tipos Temporais (data e hora)**
  - **DATE** – armazena data tipo **AAAA-MM-DD**
  - **DATETIME** – armazena data e hora no formato – AAAA- MM-DD HH:MM:SS
  - **TIME** – armazena hora no formato – **hh:mm:ss**
  - **YEAR** – armazena ano no formato – **aaaa**
  - **TIMESTAMP** – armazena data e hora podendo ser usando com TIME_STAMP

SQL – DDL

- **Tipos Alfanuméricos (textuais)**
  - **CHAR** – até 255 caracteres
  - **VARCHAR** – até 65535 caracteres
  - **TEXT –** até 65535 caracteres
  - **LONGTEXT** – até 4.294.967.295 caracteres
  - **BLOB –** objetos binários de tamanho até 65535 bytes
  - **ENUM –** String que pode conter apenas um valor ou zero
  - **SET –** String que pode conter zero ou mais valores
  - Os tipos TEXT e BLOB se diferenciam apenas pelo fato do primeiro não ser **sensível ao caso**

SQL – DDL

SQL Padrão (ANSI)

```
CHAR(tamanho)
CHARACTER(tamanho)

INT
INTEGER
SMALLINT

NUMERIC(p
DECIMAL
DEC(prec

FLOAT(prec
REAL
DOUBLE PRECISION
```

SQL2 = Padrão +

```
VARCHAR(taman
CHAR VARY
CHARA
                (tamanho)
          CHARACTER(tamanho)
          ING(tamanho)

BIT(tamanho)
BIT VARYING(tamanho)

DATETIME
TIME(precisão)
TIMESTAMP(precisão)
INTERVAL
```

IMPORTANTÍSSIMO!!!

CADA SGBD PODE TER PALAVRAS-CHAVE DIFERENTES PARA REFERENCIAR UM MESMO TIPO DE DADOS

SQL – DDL

- Restrições de Domínio
  - **NOT NULL**
    - Restrição aplicada a colunas cujos valores não podem ser nulos
  - Valores **DEFAULT**
    - Usado para inicializar o valor de uma coluna
    - Cláusula **DEFAULT <valor>** logo após a restrição:

valor *default*

```
CREATE TABLE CURSO (
 COD_CURSO INT NOT NULL DEFAULT 0, ... );
```

Restrição

SQL – DDL

- Integridade Referencial
  - Cuidados
    - Quando colunas são excluídas ou alteradas
    - Quando o valor do atributo da chave estrangeira é modificado na tabela referenciada
  - Ações disparadas quando ocorrem violações:

SET NULL
CASCADE

ON DELETE

ON UPDATE

SQL – DDL

- Integridade Referencial

```
CREATE TABLE ALUNO (
 COD_MAT_ALUNO BIGINT NOT NULL PRIMARY KEY,
   ...
 FOREIGN KEY (COD_CURSO) REFERENCES CURSO
(COD_CURSO) {ON DELETE [ CASCADE | SET NULL]};
);

CREATE TABLE ALUNO (
 COD_MAT_ALUNO BIGINT NOT NULL PRIMARY KEY,
   ...
 FOREIGN KEY (COD_CURSO) REFERENCES CURSO
(COD_CURSO) {ON UPDATE [ CASCADE | SET NULL]};
);
```

SQL – DDL

CREATE[**TABLE** | **INDEX** | **VIEW** | **TRIGGER** | **PROCEDURE** | **FUNCTION**]<nome_do_objeto>

(definições específicas conforme o tipo de objeto **);**

CREATE TABLE <nome_da_tabela>

- Colunas são especificadas primeiro, com a sintaxe:

(C_1 D_1, C_2 D_2, ..., C_n D_n,

- Depois Chaves, integridade referencial e restrições de integridade

PRIMARY KEY <lista_de_colunas_da_PK>,

FOREIGN KEY (<nome_da_coluna>) **REFERENCES** <nome_da_tab_ref>(<nome_da_coluna_ref>)**);**

  - cada C_i é uma coluna no esquema da tabela
  - D_i é o tipo de dado no domínio da coluna C_i

SQL – DDL

- Comandos para alteração de objetos de um esquema ou banco de dados

Incluir chave primária

```
ALTER TABLE <nome_tabela> ADD PRIMARY KEY (<coluna ou
  lista de colunas>);
```

Incluir chave estrangeira

```
ALTER TABLE <nome_tabela_lado_N> ADD FOREIGN KEY (<nome
  coluna>) REFERENCES <nome_tabela_lado_1>
  (<nome_coluna_chave_primaria>);
```

Alterar o nome de uma tabela

```
ALTER TABLE <nome_tabela> RENAME TO <novo_nome_tabela>;
```

SQL – DDL

- continuação...

Adicionar uma nova coluna após a última coluna da tabela

```
ALTER TABLE <nome_tabela> ADD COLUMN <nome_coluna>
  <tipo de dados> [<atributos>];
```

Adicionar uma nova coluna após uma determinada coluna

```
ALTER TABLE <nome_tabela> ADD COLUMN <nome_coluna>
  <tipo de dados> AFTER <nome_coluna_ANTERIOR>;
```

Adicionar uma nova coluna antes da primeira coluna

```
ALTER TABLE <nome_tabela> ADD COLUMN <nome_coluna>
  <tipo de dados> FIRST;
```

SQL – DDL

- continuação...

Renomear uma coluna

```
ALTER TABLE <nome_tabela> CHANGE <nome_coluna_atual>
  <nome_coluna_novo> <tipo de dados>;
```

Alterar o tipo de dados de uma coluna

```
ALTER TABLE <nome_tabela> CHANGE <nome_coluna>
  <nome_coluna> <tipo de dados>;
```

```
ALTER TABLE <nome_tabela> MODIFY <nome_coluna> <tipo de
  dados>;
```

Excluir uma coluna da tabela

```
ALTER TABLE <nome_tabela> DROP COLUMN <nome_coluna>;
```

SQL – DDL

- continuação...

Excluir uma restrição de integridade da tabela

```
ALTER TABLE <nome_tabela> DROP <foreign | primary> KEY
  <nome_constraint>;
```

SQL – DDL

- Excluir objetos de um esquema ou banco de dados

```
DROP <tipo_do_objeto> <nome_do_objeto>;
```

- Renomear tabelas de um esquema ou banco de dados

```
RENAME TABLE <nome_atual_tabela> TO <novo_nome_tabela>;
```

- Truncar tabelas de um esquema ou banco de dados

```
TRUNCATE TABLE <nome_tabela>;
```

INSTITUTO FEDERAL
CATARINENSE

Obrigado!