Banco de Dados - SQL

Eduardo Arruda

Eduardo Arruda

- Linguagem SQL
  - Estruturação das Bases de Dados Relacionais
  - Administração dos SGBDs Relacionais
    - MySQL
    - Access
    - .....
  - Comando SELECT (Projeção de Dados)

SQL - Introdução

- SQL-Structured Query Language (Linguagem de Consulta Estruturada)
  - Apesar do QUERY, não é apenas de consulta (inclusão, alteração,...)
- É fundamentada na álgebra relacional, inclui comandos para:
  - Definição, Consulta e Atualização de dados
- Histórico:
  - Definição da1ª versão em 1974 – IBM – chamada SEQUEL
  - 1975 implementado o 1º protótipo
  - Revisada e ampliada entre 1976/77.
    - **Teve seu nome alterado para SQL por razões Jurídicas**
  - Publicada como padrão para SGBDR em 1986 pela ANSI (American National Standar Institute)
    - **ANSI eqüivale a nossa ABNT**
    - **Mesmo padronizada, existem variações**
  - Versões posteriores a de 86 → SQL2 e SQL3

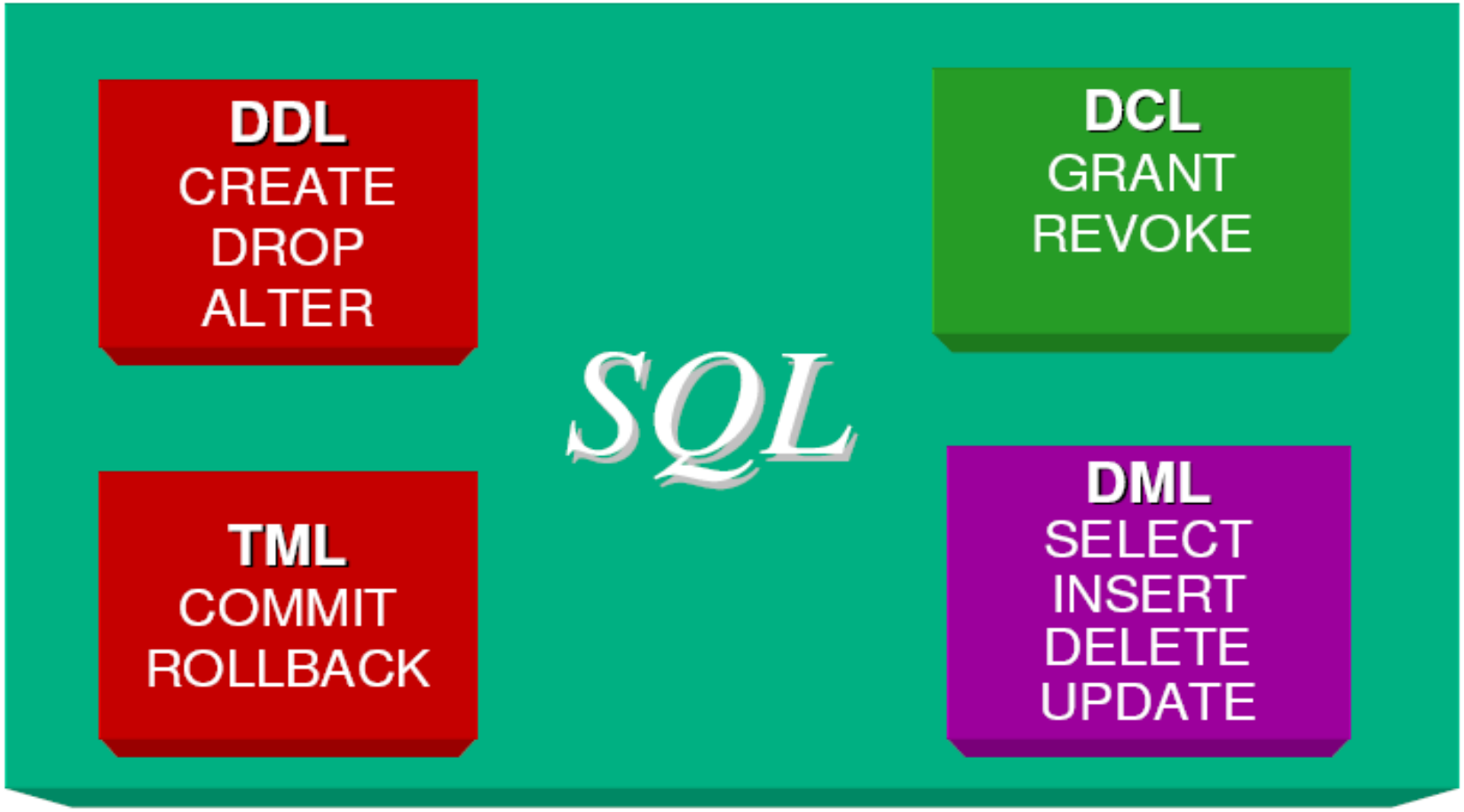
SQL - Propriedades

- Permitir consultas interativas (*query AdHoc*)
  - Usuários podem definir consultas poderosas sem a necessidade da criação de programas.
- Permite acesso e compartilhamento de dados em SGBDR
  - Pode ser embutida em programas de aplicação.
  - Pode ser usada para compartilhar dados Cliente/Servidor
- Possui comandos para administração do BD
  - O responsável pela administração do banco de dados (*BDA*) pode utilizar SQL para realizar suas tarefas.
- Maximiza a interoperabilidade entre SGBDR Heterogêneos
  - A padronização de SQL aumenta a portabilidade entre diferentes SGBDR.

SQL - Funções

- SQL provê suporte a várias funções de um SGBD :
  - **DDL** (linguagem de definição de dados)
    - **Define as tabelas (virtuais ou não) onde os dados serão armazenados.**
  - **DML** (linguagem de manipulação de dados)
    - **Permite a inclusão, remoção, atualização e seleção dos dados;**
  - **DCL** (linguagem de controle de dados)
    - **Controla o acesso e os privilégios dos usuários, protegendo os dados de manipulações não autorizadas;**
  - **TML** (linguagem de manipulação de transações)
    - **Especifica as transações, garantindo o compartilhamento e a integridade dos dados.**

SQL - Funções



A TML não será abordada neste curso !

SQL - Atenção !

- Cada implementação de SQL possui algumas adaptações para resolver certas particularidades, portanto, qualquer comando mostrado neste curso pode ser usado de forma diferente em um determinado SGBD.
  - Recomenda-se a leitura do manual do fabricante para maiores informações sobre o uso da linguagem SQL em SGBDs comerciais.
- O SQL usado neste curso será o baseado no Padrão ANSI e nenhuma característica específica de SGBD será abordada
- A maioria dos SGBDR baseiam-se no SQL ANSI

Exemplo Modelos ER & Relacional

AUTOR(CodAutor, Nome, Nascimento)

LIVRO(Titulo,CodAutor,CodEditora,Valor,Publicacao,Volume,Idioma)

CodAutor referencia AUTOR

CodEditora referencia Editora

EDITORA (CodEditora, Razao, Endereco, Cidade)

SQL - Tipos de Dados Genéricos

- Cada SGBD tem um conjunto próprio de tipos de dados.
- Genericamente estes são:
  - **Char(X):** Para dados caracteres, onde X é o tamanho máximo permitido (tamanho fixo)
  - **Varchar(X):** Idem o anterior, mas o tamanho é variável.
  - **Integer :** Para dados numéricos inteiros positivos ou negativos
  - **Decimal(X,Y):** Pada dados numéricos decimais, onde X é o tamanho máximo permitido da parte inteira e Y é o tamanho máximo da parte fracionária
  - **Date:** Para datas. Seu formato depende do SGBDR
  - **Logical:** Para os valores lógicos TRUE ou FALSE.

DDL - Criando Tabelas

- **CREATE TABLE** - Cria uma nova tabela com seus campos e define as restrições de campo.

```
CREATE TABLE Tabela (
Coluna1 Tipo [(Tamanho)] [NOT NULL] [DEFAULT] [...],
[,Coluna2 Tipo [(Tamanho)] [NOT NULL] [DEFAULT] [...],
[PRIMARY KEY (Primária1[, Primária2 [, ...]])]
[UNIQUE (Candidata1[, Candidata2[, ...]])]
[FOREIGN KEY (Estrangeira1[, Estrangeira2 [, ...]]) REFERENCES
    TabelaExterna [(ColunaExterna1 [, ColunaExterna2 [, ...]])]
[CHECK (condição)]
)
```

DDL - Criando Tabelas

- Exemplo:

```
/* Cria tabela autor */
CREATE TABLE AUTOR(
        CodAutor INTEGER NOT NULL,
        Nome CHAR(30) NOT NULL,
        Nascimento DATE NOT NULL,
        PRIMARY KEY (CodAutor),
        UNIQUE (Nome,  Nascimento) );

        /* Cria tabela editora */
        CREATE TABLE EDITORA(
            CodEditora INTEGER NOT NULL,
            Razao CHAR(30),
            Endereco CHAR(30),
            Cidade CHAR(30)
            PRIMARY KEY(CodEditora ));
```

DDL - Criando Tabelas

LAUREATE BRASIL

- Exemplo:

```
/* Cria tabela livro */
CREATE TABLE LIVRO(
    Titulo CHAR(30) NOT NULL,
    CodAutor INTEGER NOT NULL,
    CodEditora INTEGER NOT NULL,
    Valor DECIMAL(3.2),
    Comentario LOGICAL,
    Publicacao DATE,
    Volume INTEGER,
    Idioma CHAR (15) DEFAULT = 'Português',
    PRIMARY KEY (Titulo, CodAutor),
    FOREIGN KEY (CodAutor) REFERENCES AUTOR,
    FOREIGN KEY (CodEditora) REFERENCES EDITORA,
    CHECK Valor > 10.0);
```

DDL - Alterando Tabelas

- **ALTER TABLE** - permite inserir/eliminar/modificar colunas nas tabelas já existentes

ALTER TABLE *Tabela*
{ADD (*NovaColuna NovoTipo* [BEFORE *Coluna*] [, ...]) |
DROP (coluna [, ...]) |
MODIFY (*Coluna NovoTipo* [NOT NULL] [, ...]) }

Onde : | Indica escolha de várias opções
{ } Indica obrigatoriedade de escolha de uma opção entre várias

- **OBS:**
  - A cláusula DROP não remove atributos da chave primária
  - Não se usa NOT NULL juntamente com ADD, quando a tabela já contém registos (a nova coluna é carregada com NULL's)
  - Quando se altera o tipo de dados de uma coluna, os dados são convertidos para o novo tipo.
  - Se diminuir o tamanho de colunas do tipo CHAR, os dados são truncados

DDL - Alterando Tabelas

- **Exemplo:**

```
/* Adicionar o campo E-MAIL na tabela Autor */
ALTER TABLE AUTOR
ADD  EMAIL CHAR(30);

/* Modificar o campo E-MAIL na tabela Autor */
ALTER TABLE AUTOR
MODIFY EMAIL CHAR(25);

/* ELIMINAR o campo E-MAIL na tabela Autor */
ALTER TABLE AUTOR
DROP EMAIL;
```

DDL - Criando Índices

- **CREATE INDEX** - Cria um novo índice em uma tabela existente.

  CREATE [UNIQUE] INDEX *Indice* ON
  ***Tabela* (*Coluna1* [, *Coluna2*...])**

  - Os índices são estruturas que permitem agilizar a busca e ordenação de dados em tabelas

- Exemplo

```
/* Criar índice do campo nascimento
da tabela autor */
CREATE UNIQUE INDEX RazaoIDX
ON  EDITORA (Razao);
                        /* Criar índice do campo cidade e razão
                        da tabela editora */
                        CREATE INDEX AutorEditoraIDX
                        ON  LIVRO (CodAutor,CodEditora);
```

DDL - Excluindo Tabelas e Índices

- **DROP** - Exclui uma tabela ou um índice já existente
  - CUIDADO, pois os dados também são excluídos

DROP {TABLE *tabela* | INDEX *índice*}

Exemplo:

```
/* Excluir a tabela livro */
DROP TABLE LIVRO;

/* Excluir o indice CidadeRazaoIDX da tabela editora */
DROP INDEX CidadeRazaoIDX;
```

Obrigado!

Eduardo Arruda