Bancos de Dados

Prof. Ronaldo Lopes de Oliveira

Linguagens Relacionais

SQL

SQL

- **Modos de Operação com SQL**
  - SQL interativa
    - Executada a partir de ferramenta específica que fornece ambiente de execução direta de comandos SQL
  - SQL embutida
    - Comandos SQL embutidos em uma linguagem hospedeira como C, JAVA, C++, etc
    - Comandos SQL são delimitados por comandos sinalizadores específicos como EXEC-SQL, END-SQL
    - Comandos sinalizadores depende da linguagem hospedeira (reconhecida pelo pré-compilador)
    - Variáveis contidas no programa hospedeiro podem ser referenciadas nos comandos SQL para receber valores retornados do banco de dados ou servir como origem para atualizações dos dados no banco de dados

SQL

- **TIPOS de SQL**
  - SQL não procedimental
    - Comandos SQL básicos (INSERT, UPDATE, DELETE, SELECT) orientados por conjunto de tuplas
      - tratam um conjunto de tuplas de uma ou mais relações que satisfazem determinados critérios de seleção
    - Não inclui comandos para tratamento tupla a tupla

SQL

- **TIPOS de SQL**
  - SQL procedimental
    - Inclui comandos adicionais para criação de tabelas temporárias (cursor) que permitem recuperação e tratamento tupla a tupla
      - DECLARE CURSOR
      - OPEN
      - FETCH
      - CLOSE
    - Alguns SGBDs criaram versões de SQL procedimental que incluem comandos de linguagens convencionais como comandos de repetição, condição, desvio
      - Exemplo PL-SQL da Oracle

SQL

- **Padrões SQL**
  - Primeiras padronizações: 1986 (ANSI), 1987 (ISO)
  - 1989 (ISO)
    - Integridade referencial
    - Null
    - Default
    - Check constraints
  - 1992: SQL2
    - Criação explícita de schema (CREATE SCHEMA)
    - Criação de domínio (CREATE DOMAIN)
    - Stored Procedures
    - Novos tipos de dados (time e date)
    - Tabelas temporárias
    - Tabelas derivadas da cláusula FROM
    - Restrição UNIQUE
    - UNION

SQL

- **Padrões SQL**
  - 1999: SQL3
    - Extensões para O.O (SGBD objeto-relacionais)
      - Objetos complexos (vídeo, imagem, texto)
      - Hierarquia de tabelas (herança simples)
      - Tipos de dados definidos pelo usuário
      - coleções
    - Expressões regulares de emparelhamento
    - Queries recursivas
    - Gatilhos
    - OLAP

SQL

- **Padrões SQL**
  - 2003: SQL4
    - Composto por nove partes
      - Parte 1: SQL Framework:
      - Parte 2: SQL Foundation
      - Parte 3: SQL CLI (Call-Level-Interface)
      - Parte 4: SQL-PSM (Persistent Stored Modules)
      - Parte 9: SQL-MED (Management of External Data)
      - Parte 10: SQL-OLB (Object Language Binding)
      - Parte 11: SQL-Schemata
      - Parte 13: SQL-JRT (Java Routines and Types)
      - Parte 14: SQL-XML
      - Partes 5, 6, 7, 8 e 12 não foram aproveitadas

SQL

- **Padrões SQL**
  - 2003: SQL4
    - Parte 1: SQL Framework:
      - Define estrutura do padrão e relacionamento entre as partes
      - Apresenta conceitos e definições gerais
      - Define os requisitos de conformidade
      - Atualizações que vierem a ser feitas nesta parte refletem nas demais partes
    - Parte 2: SQL Foundation
      - Parte maior e mais importante
      - Define o núcleo da linguagem
      - Inclui todas as definições da SQL1999 Foundation com as devidas correções

SQL

- **Padrões SQL**
  - 2003: SQL4
    - Parte 3: SQL CLI
      - Define um rotinas padrão para invocar dinamicamente SQL a partir de programas de aplicação
      - Consiste em 60 especificações de rotinas para:
        - » Controlar conexões a servidores SQL
        - » Alocar e deslocar recursos
        - » Executar comandos SQL
        - » Controlar terminação de transação
        - » Obter informação sobre implementaçao
      - Dispensa pré-compilação

SQL

- **Padrões SQL**
  - 2003: SQL4
    - Principais novas características
      - Novos tipos de dados: BIGINT, MULTISET
      - Extensões a tipos de dados: ARRAY ilimitado
      - Remoção de tipos de dados: BIT, BIT VARYING
      - Geração de Sequências (SEQUENCE)
      - Colunas Identidade
      - Colunas geradas (derivadas)
      - Aumento de funcionalidade de CREATE LIKE
      - Tabelas Base criadas a partir de queries
      - Funções que retornam tabelas (“TABLE” Functions)

SQL

- **Padrões SQL**
  - 2003: SQL4
    - Principais novas características (continuação)
      - SQL dinâmico e SQL-DDL dentro de rotinas (procedures, functions)
      - Novas funções para escalares: LN, EXP, POWER, SQRT, FLOOR, CEILING, WIDTH-BUCKET
      - Novas funções de agregação de um argumento: STDDEV_POP, STDDEV_SAMP, VAR_POP, VAR_SAMP
      - Novas funções de agregação com dois argumentos: COVAR, CORR, ...
      - Novas funções para tabelas e tabelas particionadas: RANK, ROWNUMBER...
      - Comando DML Merge

SQL

- SQL-DDL Comandos Principais
  - CREATE SCHEMA
  - DROP SCHEMA
  - CREATE TABLE
  - ALTER TABLE
  - DROP TABLE
  - CREATE VIEW
  - ALTER VIEW
  - DROP VIEW

SQL

- CREATE TABLE

```
CREATE TABLE Departamentos
      (codigo numeric(2) NOT NULL PRIMARY KEY,
       nome  Varchar(30) NOT NULL UNIQUE)

CREATE TABLE Cursos
      (codigo Numeric(3) NOT NULL,
       nome varchar(40) NOT NULL,
       modalidade char(1) NOT NULL
                          CHECK (modalidade IN (‘P’, ‘S’, ‘D’)).
       codigoDepartamento Numeric(2) NOT NULL DEFAULT 1,
       PRIMARY KEY (Codigo),
       FOREIGN KEY (codigoDepartamento)
                          REFERENCES Departamentos (Codigo)
                          ON DELETE SET DEFAULT
                          ON UPDATE CASCADE)
```

SQL

- SQL-DDL: Tratamento de restrições
  - CREATE TABLE
    - Podem ser especificadas ações a serem disparadas quando uma operação causar uma potencial violação através de cláusulas ***referencial triggering action ON DELETE e ON UPDATE*** com opções:
      - SET DEFAULT
      - SET NULL
      - CASCADE
  - DROP SCHEMA
    - DROP SCHEMA nome-esquema CASCADE
    - DROP SCHEMA nome-esquema RESTRICT
  - DROP TABLE
    - DROP TABLE nome-tabela CASCADE
    - DROP TABLE nome-tabela RESTRICT

SQL

- SQL-DDL: Tratamento de restrições
  - ALTER TABLE
    - Adição de atributo (coluna)
      - ALTER TABLE nome-tabela ADD nome-atributo...
        - » Se o atributo não puder ter valor NULL então deve ser especificado valor default

          Exemplo: ALTER TABLE Empregados

          ADD estCivil char(1) DEFAULT ‘S’
    - Remoção de atributo (coluna)
      - ALTER TABLE nome-tabela DROP nome-atributo
        - » Deve ser especificada opção RESTRICT ou CASCADE para tratar restrições de integridade referencia

          Exemplo: ALTER TABLE Empregados

          DROP estCivil CASCADE

SQL

- SQL-DDL: Tratamento de restrições
  - ALTER TABLE
    - Alteração de atributo (coluna)
      - Adicionar/remover valores default

        Exemplo: ALTER TABLE Empregados

        ALTER estCivil DROP DEFAULT

      - Adicionar/remover restrições

        Exemplo: ALTER TABLE Empregados

        DROP CONSTRAINT deptoFK CASCADE

SQL

- SQL-DML
  - INSERT
    - Acréscimo de múltiplos registros:

```
INSERT INTO tabdestino [(campo1[, campo2[, ...]])]
<comando-select>
Exemplo:
        INSERT INTO Empregados (nome, matricula,
                                                salario)
                SELECT * FROM EmpregadosAntigos;
```

    - Acréscimo de registro único:

```
INSERT INTO tabdestino [(campo1[, campo2[, ...]])]
VALUES (valor1[, valor2[, ...])
Exemplo:
        INSERT INTO Empregados (nome, matricula,
salario)
        VALUES ('Ronaldo Lopes',3523,5000);
```

SQL

- SQL-DML
  - UPDATE

```
UPDATE tabela
SET campo1 = exprValor [,campo2 = exprValor...]
[WHERE condições]
```

Exemplo:

```
UPDATE Pedidos
SET     QuantiaDoPedido = QuantiaDoPedido * 1.1,
        Frete = Frete * 1.03
WHERE PaísDeDestino = 'UK';
```

SQL

- SQL-DML
  - DELETE

    DELETE FROM tabela
    [WHERE condições]

    Exemplo:

    ```
    DELETE FROM Pedidos
    WHERE PaísDeDestino = 'UK';
    ```

SQL

- SQL-DML: Tratamento de restrições
  - DELETE
    - O tratamento das restrições de integridade referencial de tupla(s) de outra(s) tabela(s) que referencia(m) a(s) tupla(s) removida(s) segue o que foi especificado no comando de criação da tabela que referencia.
  - UPDATE
    - O tratamento das restrições de integridade referencial de tupla(s) de outra(s) tabela(s) que referencia(m) a(s) tupla(s) atualizada(s) segue o que foi especificado no comando de criação da tabela que referencia.

SQL

- SQL-DQL
  - SELECT

    Sintaxe básica:

    SELECT <lista-atributos>
    FROM <lista-tabelas>
    [WHERE <condições>]
    [GROUP BY <atributos-grupo>]
    [HAVING <condições-grupo>]
    [ORDER BY <lista-atributos-ordenação]

SQL

- SQL-DQL
  - SELECT – Variantes sintáticas importantes
    - Convenção *
    - Tratamento de ambiguidade de nomes
    - Cláusula DISTINCT
    - Operações de conjunto
      - UNION [ALL]
      - INTERSECT [ALL]
      - EXCEPT [ALL]
    - Ordenação ascendente e descendente

SQL

- SQL-DQL
  - SELECT – Variantes sintáticas importantes
    - Comparações de strings
      - LIKE
        - » Convenção %
        - » Convenção _
        - » Convenção \
      - Concatenação (Convenção ||)
    - Comparações numéricas
      - Operadores de comparação (<, >, <>, =)
      - Between
    - Pertinência de valor de atributo em conjunto (IN, NOT IN)
    - Comparação com NULL (IS NULL, IS NOT NULL)

SQL

- SQL-DQL
  - SELECT – Variantes sintáticas importantes
    - Funções de tratamento de tipos de dados específicos (*date, time, strings, timestamp, numeric*, etc)
      - Em geral cada SGBD implementa seu subconjunto particular de funções e procedimentos embora com semânticas similares
    - Consultas aninhadas (subconsultas)
      - Verificar se o valor de um ou mais atributos da consulta externa está contido no conjunto de valores retornados por uma consulta interna

        (SELECT... FROM...WHERE atributo IN/NOT IN (SELECT ...)
      - Comparação de valor de atributo com valor(es) retornados pela subconsulta

        (SELECT... FROM...WHERE atributo > ALL (SELECT ...)

SQL

- SQL-DQL
  - SELECT – Variantes sintáticas importantes
    - Consultas aninhadas (subconsultas)
      - Correlacionar atributos da consulta externa com atributos da consulta interna

Exemplo

```
SELECT e.nome
FROM empregados AS e
WHERE e.mat in (SELECT d.mat
                FROM Dependentes AS d
                WHERE d.sexo = e.sexo)
```

SQL

- SQL-DQL
  - SELECT – Variantes sintáticas importantes
    - EXISTS / NOT EXISTS

Exemplo

```
SELECT e.nome
FROM empregados AS e
WHERE EXISTS (SELECT d.mat
              FROM Dependentes AS d
              WHERE d.sexo = e.sexo)
```

SQL

- SQL-DQL
  - SELECT – Variantes sintáticas importantes
    - Tabelas de Junção
      - Incorporado na SQL2
      - Permite a inclusão de condição de junção diretamente na cláusula FROM ao invés de incluí-las na cláusula WHERE
      - Facilita a incorporação de variantes de junção (outer join, natural join)

Exemplo:

```
SELECT  e.nome AS 'Nome Empregado',
                    d.nome AS 'Departamento'
FROM
      Depto AS d LEFT OUTER JOIN Emp AS e ON
                                          d.codigo=e.depto)
```

SQL

- SQL-DQL
  - SELECT – Variantes sintáticas importantes
    - Funções agregadas e agrupamento
      - COUNT
      - SUM
      - MAX
      - MIN
      - AVG
      - AVGP
      - DEV
      - DEVP

Exemplo:

```
SELECT depto, max(sal), min(sal), avg(sal)
FROM    empregados
WHERE sexo = 'M'
GROUP BY depto
HAVING count(*) > 10;
```

SQL

- SQL-DTL
  - BEGIN (TRANSACTION/WORK)
  - COMMIT
  - ROLLBACK
- SQL-DCL
  - CREATE USER
  - ALTER USER
  - DROP USER
  - GRANT
  - REVOKE
  - CREATE SYNONYM

SQL

- Padrões de Conectividade
  - API para conectividade com banco de dados
  - Permite que programas em linguagens convencionais enviem instruções SQL através de chamadas padronizadas para qualquer banco de dados que disponibilize *driver* que implemente a API
  - Nesta abordagem, a aplicação, em tempo de execução:
    - Especifica qual a fonte de dados a que deseja ter acesso.
    - Faz a vinculação entre a aplicação e a fonte de dados, através de um módulo (driver).
    - O driver converte os formatos de dados e os comandos padronizados para os formatos compreendidos pelo SGBD-alvo.

SQL

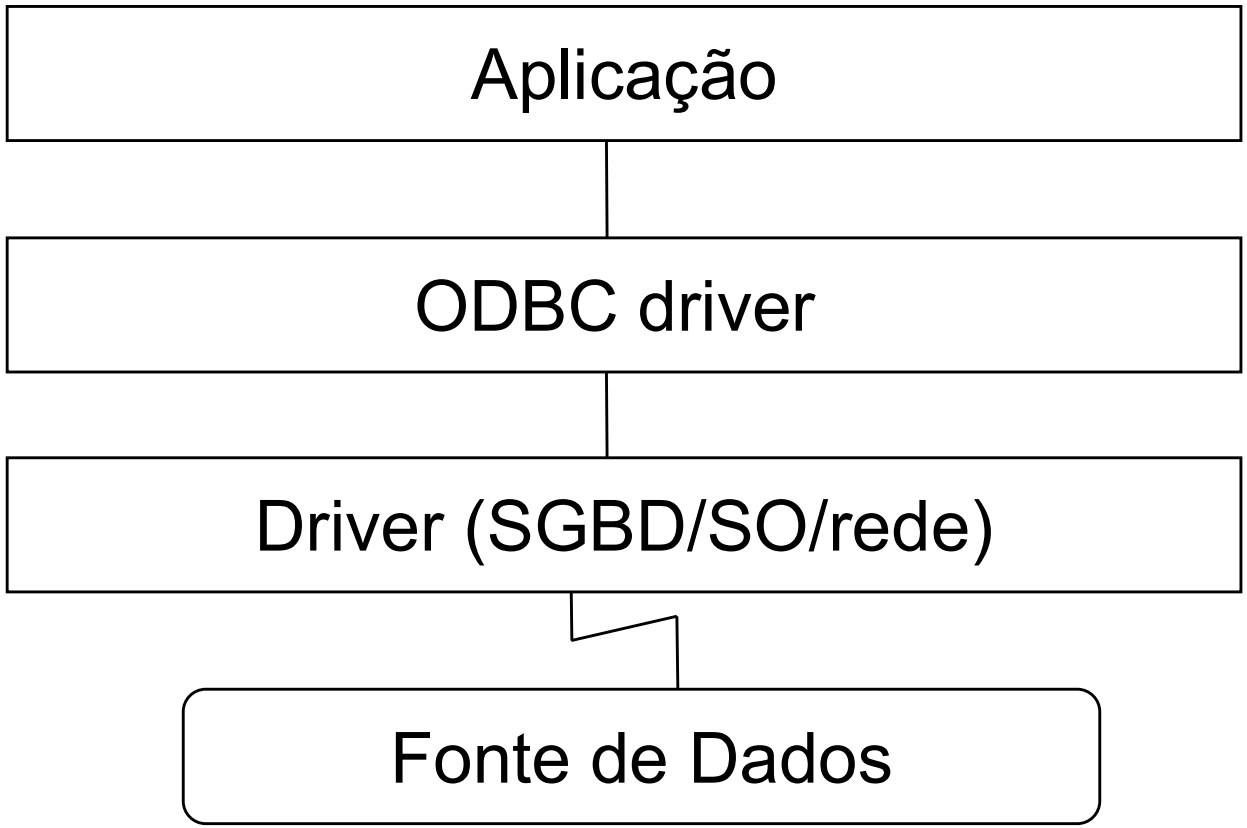
- Padrões de Conectividade
  - Vantagens
    - Não é necessário utilizar várias linguagens para acesso a dados de diferentes SGBDs
      - Menor exigência de treinamento dos desenvolvedores
      - Menos erros de programação
      - Mais rápido desenvolvimento
      - Os desenvolvedores não precisam se preocupar com as particularidades dos bancos de dados que irão acessar e trabalhar.
  - Desvantagens
    - Necessidade de mais camadas de processamento
    - Desempenho

SQL

- ODBC (Open Database Connectivity):
  - Criado no início da década de 1990 pelo SQL Access Group liderado pela Microsoft
  - Através de chamadas ODBC em um programa podem ser acessados diferentes bancos de dados sem utilizar as interfaces proprietárias de cada um.
  - Driver para cada SGBD alvo conecta dinamicamente uma biblioteca à aplicação
  - Driver máscara a heterogeneidade de SGBD, sistema operacional e protocolo de rede.
    - Exemplo (Sybase, Windows/NT, TCP/IP driver)

SQL

Arquitetura ODBC



SQL

- JDBC (Java Database Connectivity)
  - API Java para conectar programas escritos em Java com bancos de dados relaconais.
  - Composto por um conjunto de classes e interfaces escritos em Java
  - Padrão definido pela Sun Microsystems
  - Permite que fornecedores de soluções em bancos de dados, implementem e extendam o padrão usando seus próprios *drivers* JDBC.
  - Permite aos programadores Java conectar-se a bancos de dados e a acessá-lo e manipulá-lo utilizando SQL.

SQL

Arquitetura JDBC