



Universidade Federal de Sergipe
Departamento de Sistemas de Informação
Itatech Group Jr – Softwares Itabaiana

Site: www.itatechjr.com.br

E-mail: contato@itatechjr.com.br



Linguagem SQL (Parte I)

Introdução a Banco de Dados

André Vinicius R. P. Nascimento
andreviniciusnascimento@gmail.com



Conteúdo

- A Linguagem SQL
- Microsoft Sql Server
- Ferramentas
- Criando Tabelas
- Comando INSERT
- Comando UPDATE
- Comando DELETE
- Comando SELECT



A Linguagem SQL

- O que é SQL ?
 - Linguagem de consulta declarativa para manipulação de dados em SGBDs (Sistemas Gerenciadores de Banco de Dados) Relacionais.



A Linguagem SQL

- O que é ANSI-SQL ?
 - Especificação feita pela ANSI em 1986.
 - SQL1 ou ANSI 1986
 - ANSI 1989
 - SQL2 ou SQL92
 - SQL3 ou SQL99.
 - SQL 2003
 - SQL 2006
 - SQL 2008



A Linguagem SQL

- Dialetos SQL

- A linguagem SQL possui diversos dialetos pertencentes a vários produtos.
- Comandos desses dialetos podem apresentar diferenças na sintaxe ou na semântica, sendo que a última é bem mais rara.

- O que é Transact-SQL ou T-SQL?

- Dialeto da Linguagem SQL utilizado pelo SQL Server.
- Utilizaremos esse dialeto para estudar a linguagem SQL.



Microsoft SQL Server

- O que é o SQL Server ?
 - Sistema gerenciador de banco de dados relacional (RDBMS)
 - Responsável por :
 - Manter os relacionamentos entre os dados de um banco de dados;
 - Garantir integridade dos dados;
 - Garantir recuperação do banco de dados para um estado consistente em caso de falhas de sistema.



Microsoft SQL Server

- Principais Ferramentas:
 - Versão 2000
 - Enterprise Manager
 - SQL Query Analyser (isqlw)
 - Versões 2005 e 2008
 - SQL Server Management Studio



Microsoft SQL Server

- Bancos de Dados SQL Server
 - É uma Coleção de Objetos que armazenam e manipulam dados
 - Possui seu próprio conjunto de tabelas de sistema que armazenam a definição do banco de dados
 - Possui um log de Transações



Microsoft SQL Server

- Bancos de Dados SQL Server
 - Fisicamente, é composto por dois ou mais arquivos.
 - O SQL Server admite os seguintes tipos de arquivos: Primário, Secundário e Arquivo de Log.



Microsoft SQL Server

- Bancos de Dados SQL Server
 - Primary Data Files: Todo banco de dados possui um arquivo primário que, além de armazenar dados, gerencia ou outros arquivos (.MDF)
 - Secondary Data Files: Um banco de dados pode possuir 0 ou mais arquivos secundários (.NDF)
 - Log Files: Todo banco de dados possui um ou mais arquivos de log (.LDF)



Microsoft SQL Server

- Bancos de Dados Padrão do SQL Server
 - Master – Guarda Informações sobre os outros bancos de dados.
 - Model - Modelo para a criação dos novos bancos de dados.
 - Msdb – Usado pelo serviço SQL Server Agent para agendamentos e o histórico de serviços.
 - Tempdb – Banco de dados temporário usado para classificações, associações e outros processos que necessitam de espaço temporário.



Criando Tabelas

- Tipos de dados

- Toda coluna, variável, parâmetro possui um tipo de dado associado.
- No Sql Server temos vários tipos de dados definidos pelo sistema (pré-definidos).



Criando Tabelas

- Tipos de dados fornecidos pelo sistema
 - Inteiro : bigint, int, smallint, tinyint
 - Numérico exato : numeric, decimal
 - Numérico aproximado : float, real
 - Data e Hora : datetime, smalldatetime
 - Caracter : char, varchar, text
 - Caracter Unicode : nchar, nvarchar, ntext
 - Binário : binary, varbinary



Criando Tabelas

- O Comando Create Table
 - Utilizamos o comando CREATE TABLE para criar tabelas no banco de dados
 - Definição de colunas e tipos de dados
 - Definição de restrições (NULL, NOT NULL)
 - Definição de valores padrão



Criando Tabelas

```
CREATE TABLE Tb_Cliente (  
    cd_cliente int NOT NULL,  
    nm_cliente varchar(50) NOT NULL,  
    rua varchar(50) NOT NULL,  
    bairro varchar(50) NOT NULL,  
    telefone varchar(15) NULL,  
    dt_nascimento datetime,  
    cd_titular int NULL,  
    dt_inclusao    datetime default (getdate()),  
)
```



Criando Tabelas

- As colunas de uma Tabela podem ser campos calculados

```
CREATE TABLE Tb_Estoque (  
    cd_produto int,  
    qt_produto int,  
    vl_unitario numeric(15,2),  
    vl_estoque AS (qt_produto * vl_unitario)  
)
```



Criando Tabelas

- As colunas de uma Tabela podem possuir a propriedade autoincremento

```
Create Table Tb_Categoria_Fita (  
    cd_categoria int identity(1,1) NOT NULL,  
    nm_categoria varchar(50) NOT NULL,  
    valor_diario numeric(15,2)  
)
```



Criando Tabelas

- `IDENTITY(SEMENTE, INCREMENTO)`
- Somente para os tipos de dados: `bigint`, `int`, `smallint`, `tinyint`, e os tipos `decimal` e `numeric` com escala 0.
- Somente uma coluna por tabela.



Criando Tabelas

- Adicionando uma coluna a uma tabela

```
ALTER TABLE TB_ESTOQUE ADD NM_PRODUTO  
    VARCHAR(40) NULL
```

- Adicionando mais de uma coluna a uma tabela

```
ALTER TABLE TB_ESTOQUE ADD  
NM_PRODUTO VARCHAR(40) NULL,  
QT_MINIMA_ESTOQUE NUMERIC(5) NOT NULL
```



Criando Tabelas

- Alterando uma coluna de uma tabela

```
ALTER TABLE TB_ESTOQUE ALTER COLUMN  
NM_PRODUTO VARCHAR(50) NULL
```



Criando Tabelas

- Removendo uma coluna de uma tabela

```
ALTER TABLE TB_ESTOQUE DROP COLUMN  
NM_PRODUTO
```

- Removendo uma Tabela

```
DROP TABLE TB_ESTOQUE
```



Comando Insert

- Considere as tabelas

```
CREATE TABLE Tb_Loja (  
    cd_loja int NOT NULL PRIMARY KEY,  
    nm_loja varchar (40) NOT NULL,  
    categoria int NULL,  
    estado char(2) default('SE')  
)
```

```
CREATE TABLE Tb_Departamento (  
    cd_departamento int identity(1,1) PRIMARY KEY,  
    nm_departamento varchar(40) NOT NULL,  
    cd_loja int NULL,  
)
```



Comando Insert

- Utilizado para inserir valores em uma tabela

```
INSERT INTO TB_LOJA(cd_loja, nm_loja, categoria,  
estado)  
VALUES (1, 'LOJA SUL', 2, 'SE')
```

```
INSERT INTO TB_LOJA(cd_loja, nm_loja, categoria)  
VALUES (1, 'LOJA SUL', 2)
```



Comando Insert

- Não são informados valores para campos do tipo identity

```
INSERT INTO TB_DEPARTAMENTO (nm_departamento,  
    cd_loja)  
VALUES ('Departamento 1 LOJA SUL',1)
```



Comando Insert

- Podemos omitir a lista de colunas. Nesse caso todos os valores devem ser fornecidos na ordem em que foram definidos

```
INSERT INTO TB_LOJA  
VALUES (1, 'LOJA SUL', 2, 'SE')
```

```
INSERT INTO TB_DEPARTAMENTO  
VALUES ('Departamento 1 LOJA SUL', 1)
```



Comando Update

- Utilizado para modificar valores em uma tabela

```
UPDATE TB_LOJA  
SET NM_LOJA = 'LOJA SUDESTE'  
WHERE CD_LOJA = 1
```

```
UPDATE TB_LOJA  
SET NM_LOJA = 'LOJA SUDESTE',  
    ESTADO = 'AL'  
WHERE CD_LOJA = 1
```



Comando Delete

- Utilizado para remover linhas de uma tabela

```
DELETE FROM TB_LOJA  
WHERE CD_LOJA = 1
```

```
DELETE FROM TB_LOJA OU DELETE TB_LOJA
```

- Para remover todas as linhas, melhor utilizar o truncate

```
TRUNCATE TABLE TB_LOJA
```



Comando Select

- Utilizado para retornar um conjunto de linhas de uma ou mais tabelas

```
SELECT <LISTA DE COLUNAS>  
FROM <LISTA TABELAS>  
WHERE <CONDICAO>
```

```
SELECT CD_LOJA, NM_LOJA, ESTADO  
FROM TB_LOJA  
WHERE CD_LOJA = 1
```



Comando Select

- O * substitui todas as colunas

```
SELECT * FROM TB_LOJA
```



Comando Select

- A cláusula `distinct` é utilizada para remover linhas repetidas

```
SELECT DISTINCT ESTADO FROM TB_LOJA
```



Comando Select

- Podemos informar Aliases para as colunas da tabela

```
SELECT ESTADO AS SIGLA FROM TB_LOJA
```

- O operador AS pode ser omitido. CUIDADO !

```
SELECT ESTADO CATEGORIA FROM TB_LOJA
```



Comando Select

- Utilizamos ` ` para aliases com espaço em branco

```
SELECT ESTADO AS 'SIGLA DO ESTADO' FROM TB_LOJA
```



Comando Select

- Operadores na Condição
 - AND
 - OR
 - IN
 - NOT IN



Comando Select

- O operador LIKE
 - Permite a comparação de cadeias de caracteres com padrões

<CADEIA DE CARACTERES> LIKE <PADRÃO>

```
SELECT ESTADO FROM TB_LOJA  
WHERE NM_LOJA LIKE '%SUL%'
```

```
SELECT ESTADO FROM TB_LOJA  
WHERE NM_LOJA LIKE '%S_L%'
```



Comando Select

- O operador LIKE

```
SELECT ESTADO FROM TB_LOJA  
WHERE NM_LOJA LIKE '%S%' ESCAPE 'S'
```

```
SELECT ESTADO FROM TB_LOJA  
WHERE NM_LOJA LIKE '%S_L%'
```



Comando Select

- Ordenando Resultados

```
SELECT CD_LOJA, NM_LOJA FROM TB_LOJA  
ORDER BY NM_LOJA
```

```
SELECT CD_LOJA, ESTADO, NM_LOJA FROM TB_LOJA  
ORDER BY ESTADO DESC , NM_LOJA
```

```
SELECT CD_LOJA, NM_LOJA FROM TB_LOJA  
ORDER BY 1
```



Comando Select

- Recuperando as primeiras linhas

```
SELECT TOP 2 CD_LOJA, NM_LOJA FROM TB_LOJA
```

```
SELECT TOP 2 CD_LOJA, NM_LOJA FROM TB_LOJA  
ORDER BY NM_LOJA
```

```
SELECT TOP 50 PERCENT CD_LOJA, NM_LOJA FROM  
    TB_LOJA  
ORDER BY NM_LOJA
```



Comando Select

- União de Consultas

```
SELECT CD_LOJA, NM_LOJA FROM TB_LOJA1  
UNION
```

```
SELECT CD_LOJA, NM_LOJA FROM TB_LOJA2
```

```
SELECT CD_LOJA, NM_LOJA FROM TB_LOJA1  
UNION ALL
```

```
SELECT CD_LOJA, NM_LOJA FROM TB_LOJA2
```



Comando Insert + Select

- Podemos utilizar o resultado de uma consulta como valores para o comando insert

```
INSERT INTO TB_DEPARTAMENTO02 (nm_departamento,  
    cd_loja)  
SELECT nm_departamento, cd_loja from  
    tb_departamento
```