Lucas Santos – GUIA SQL

CONSIDERAÇÕES IMPORTANTES

Este guia foi feito com base no uso da linguagem SQL no SQL SERVER

SQL (Structured Query Language) é uma linguagem usada para se comunicar com bancos de dados relacionais. Com ela, podemos criar bancos, tabelas, inserir dados, atualizar, buscar, deletar e muito mais.

OBS -> O SQL pode ser case sensitivy ou não dependendo da IDE e da forma que a mesma foi instalada (este guia utiliza tudo em minúsculo).

OBS.2-> A tabela utilizada nos exemplos é:

```
create table alunos (
  matricula INT,
  nome VARCHAR(100),
  cpf VARCHAR(20),
  rg VARCHAR(20),
  uf_rg VARCHAR(2),
  idade INT,
  curso VARCHAR(100)
);
```

Os nomes utilizados nos exemplos podem ser alterados para o contexto de uso do programador e de suas tabelas/banco de dados.

Categorias de Comandos SQL

Os comandos em SQL são separados em 4 categorias principais.

DDL -> Data Definition Language. Cria ou altera estruturas de dados.
Exemplo: Create, alter, drop.
DML -> Data Manipulation Language. Manipula os dados.
Exemplo: Insert, Update, delete.
DQL -> Data Query Language. Consulta os dados.
Exemplo: Select.
DCL -> Data Control Language. Controla permições.
Exemplo: Grant, Revoke.

COMANDOS SQL

Comandos de Criação: DDL

1) Constraints

São restrições que garantem a qualidade e consistência dos dados nas tabelas.

primary key -> Define que um campo da tabela é a chave primaria.
not null -> Impede que um campo receba valores nulos.
unique -> Garante que os valores não se repitam na tabela.
identity -> Gera automaticamente um valor pro campo. Sua incrementação pode ter o valor definido praviamente.
Exemplo: identity(valor_inicial, incremento)
Identity(1, 1) -> começa em 1 e soma 1 a cada chamada.
default -> Define um valor padrão para um campo.
check -> Impede valores fora de uma regra.
foreign key -> Define que um campo da tabela é uma chave estrangeira vinda de outra tabela.

Exemplos:

```
create table Clientes{
        id int primary key, //chave primaria
        nome varchar(50) not null, //campo não nulo
        cpf char(11) unique, //valor único no campo
        idade int check (idade >= 18), //idade deve ser maior que 18
        cidadeID int, // chave estrangeira
        constraint FK_cidade foreign key (cidadeID) //foreign key nomeada
        references Cidades(id)
}

Create table Cidades{
        id int primary key identity(1, 1), //identity
        nome varchar(100) notnull
}
```

2) Create

São responsáveis pela criação de tabelas e bancos de dados.

create database Nome_banco -> Cria um banco de dados.
Exemplo: create database banco1 -> Cria um banco de dados chamado banco1.

create table nome_tabela -> Cria uma tabela no banco de dados.
É recomendado que todos os campos da tabela sejam criados nesse momento.

Exemplo: create table alunos{
id int identity(1,1) primary key,
nome varchar(50) not null,
cpf varchar(20) not null,
rg varchar(20),
idade int not null,
curso varchar(100) not null
}

Comandos de Seleção: DQL

São utilizados para consultar dados de tabelas. É possível retonar todos os campos de uma tabela de uma vez, só alguns campos, campos com restrições e demais outras variedades de consultas.

* -> O operador "*" é um coringa utilizado para referenciar todas as colunas de uma tabela.

select * from alunos -> Retorna todas as colunas da tabela Alunos.

select campo1, campo2 from tabela -> É possível retornar so alguns campos da da tabela.
Exemplo: select id, nome, curso from alunos -> retorna 3 campos da tabela alunos.

Where – friltragem de dados

O comando where perm Permite impor restrições as consulas à tabela utilizando funções ou operadores presentes no SQL.

Operadores de comparação

São utilziados para realizar comparações de valores entre diferentes valores e campos presentes nas tabelas.

= -> Utilizado para validar se o valor do campo é igual ao requisitado.
Exemplo: select id, nome from alunos where curso = 'ADS'
Retorna o id e nome dos alunos que cursam ADS.

!= -> Utilizado para validar se o valor do campo é diferente ao requisitado.
Exemplo: select id, nome, curso from alunos where nome != 'Lucas'
Retorna o id,nome e curso dos alunos cujo nome não é Lucas.
OBS-> também pode ser representado por: <>

> -> Utilizado para validar se o valor do campo é maior que o requisitado.
Exemplo: select id, nome, curso from alunos where idade > 18
Retorna o id,nome e curso dos alunos cujo nome não é Lucas.

< -> Utilizado para validar se o valor do campo é menor que o requisitado.
Exemplo: select id, nome, curso from alunos where idade < 50
Retorna o id,nome e curso dos alunos cujo nome não é Lucas.

>= -> Utilizado para validar se o valor do campo é maior ou igual ao requisitado.
Exemplo: select id, nome, curso from alunos where id >= 30
Retorna o id,nome e curso dos alunos cujo nome não é Lucas.

<= -> Utilizado para validar se o valor do campo é menor ou igual ao requisitado.
Exemplo: select id, nome, curso from alunos where idade >= 18
Retorna o id,nome e curso dos alunos cujo nome não é Lucas.

Operadores Lógicos

São utilizados para realizar combinações de múltiplas condições difentes dentro de um mesma propriedade where.

and -> Operador “E”. Utizado para validar se todas as condições passadas são verdadeiras.

Exemplo: select id, nome, cpf from alunos where idade >= 18 and curso = ‘ADS’

Verifica se a idade do aluno é maior ou igual a 18 E se o curso do aluno é ADS. Caso as duas condições sejam verdadeiras, ele retorna a linha.

or -> Operador “OU”. Utilizado para validar se qualquer uma das condições é verdadeira.

Exemplo: select id, nome, cpf from alunos where curso = “Letras” or idade < 50.

Verifica se o curos do aluno é Letras ou a idade dele é menor que 50. Caso qualquer uma das condições seja verdadeira, ele retorna a linha.

not -> Operador “Negação”. Utilizado para validar a negação de uma condição.

Exemplo: select nome, curso from alunos where not(idade < 18)

Verifica se a idade do aluno é menor que 18. Caso não, ele retorna o aluno, que terá idade >= 18.

Operadores de Padrão e Faixa

São operadores especiais que trabalham com padrões, listas ou intervalos. Também aparecem geralmente em clausulas where.

between -> Operador “entre”. Utilizado para verificar se o valor está dentro de um intervalo.

Exemplo: select * from alunos where id between 1 and 20

Verifica se o id do aluno está entre 1 e 20. Caso sim, retorna a linha.

in -> Operador “Dentro”. Verifica se o valor está dentro de uma lista.

Exemplo: select id, nome, curso from alunos where curso in(‘ADS’, ‘Letras’, ‘Fisica’)

Verifica se o aluno está em um dos 3 cursos. Caso sim, retorna a linha.

OBS: Alternativa para não usar excessivamente os operadores and e or.

is null -> Utilizado para verificar se o valor do elemento é nulo.

Exemplo: select * from alunos where rg is null

Verifica se o campo rg é nulo. Caso sim, retorna a linha

is not null -> Utilizado para verificar se o valor do elemento não é nulo.

Exemplo select * from alunos where rg is not null

Verifica se o campo rg não é nulo. Caso não seja, ele retorna a linha.

like -> Utilizado para comparar string com padrões utilizando os curingas _ e %.

_ -> Cada _ substitui um caractere na string que se deseja avaliar.

% -> Cada % substitui qualquer sequência de caracteres(incluindo espaços).

Exemplos de _:

select * from alunos where nome like ‘A__’

Exibe todas as linhas cujos nomes tem 3 letras e começam com ‘A’(Ana, Ari, Ada etc).

select * from alunos where nome like ‘_____’

Exibe todas as linhas cujos nomes tem 5 letras(Jorge, Lucia, Lucas etc).

select * from alunos where nome like ‘_u_a_’

Exibe todas as linhas cujos nomes tem 5 letras e a segunda e quarta letra são u e a. (Lucas).

Exemplos de %:
select * from alunos where nome like ‘L%’
Exibe todas as linhas cujos nomes começam com L (Lucas, Lucia, Laura etc).

select * from alunos where nome like ‘%s’
Exibe todas as linhas cujos nomes terminam com s (Lucas, Agnes, Marcos etc).

select * from alunos where nome like ‘%a%’
Exibe todas as linhas cujos nomes possuam a letra a (Paula, Lucas, Angela etc).

select * from alunos where nome like ‘%S_t_o_%’
Exibe todas as linhas cujos nomes possuan a sequencia S_t_o_ de caracteres (Lucas Santos, Silvio Santos, Alberto Santos Dummnt etc).

Clausula order by

Responsavel por ordenar os resultados por um ou mais campos da tabela, podendo definir a ordem da tabela (crescente ou decrescente).

asc -> Faz com que a consulta retorne os valores em ordem ascendente, ou seja, do menor para o maior.
Exemplo: select * from alunos order by idade asc
Exibe todas as linhas da tabela alunos de forma ascendente em idade (da menor idade em primeiro até a maior no fim).

desc -> Faz com que a consulta retorne os valores em ordem decrescente, ou seja, do maior para o menor.
Exemplo: select * from alunos order by idade desc
Exibe todas as linhas da tabela alunos de forma decrescente em idade (da maior idade em primeiro até a menor no fim).

PROPRIEDADES/FUNÇÕES