

# INTRODUÇÃO A SQL

DEPARTAMENTO DE CIÊNCIAS EXATAS E DA TERRA, CAMPUS I

PROFESSORA: **Daniela Barreto Araújo**

09/2020

# AGENDA

Objetivos de aprendizagem  
Conhecimentos prévios/contextualização  
Principais comandos DML  
Atividade extraclasse  
Referências

# OBJETIVOS DE APRENDIZAGEM

1. Reconhecer os principais comandos DML.

# PRINCIPAIS COMANDOS DML

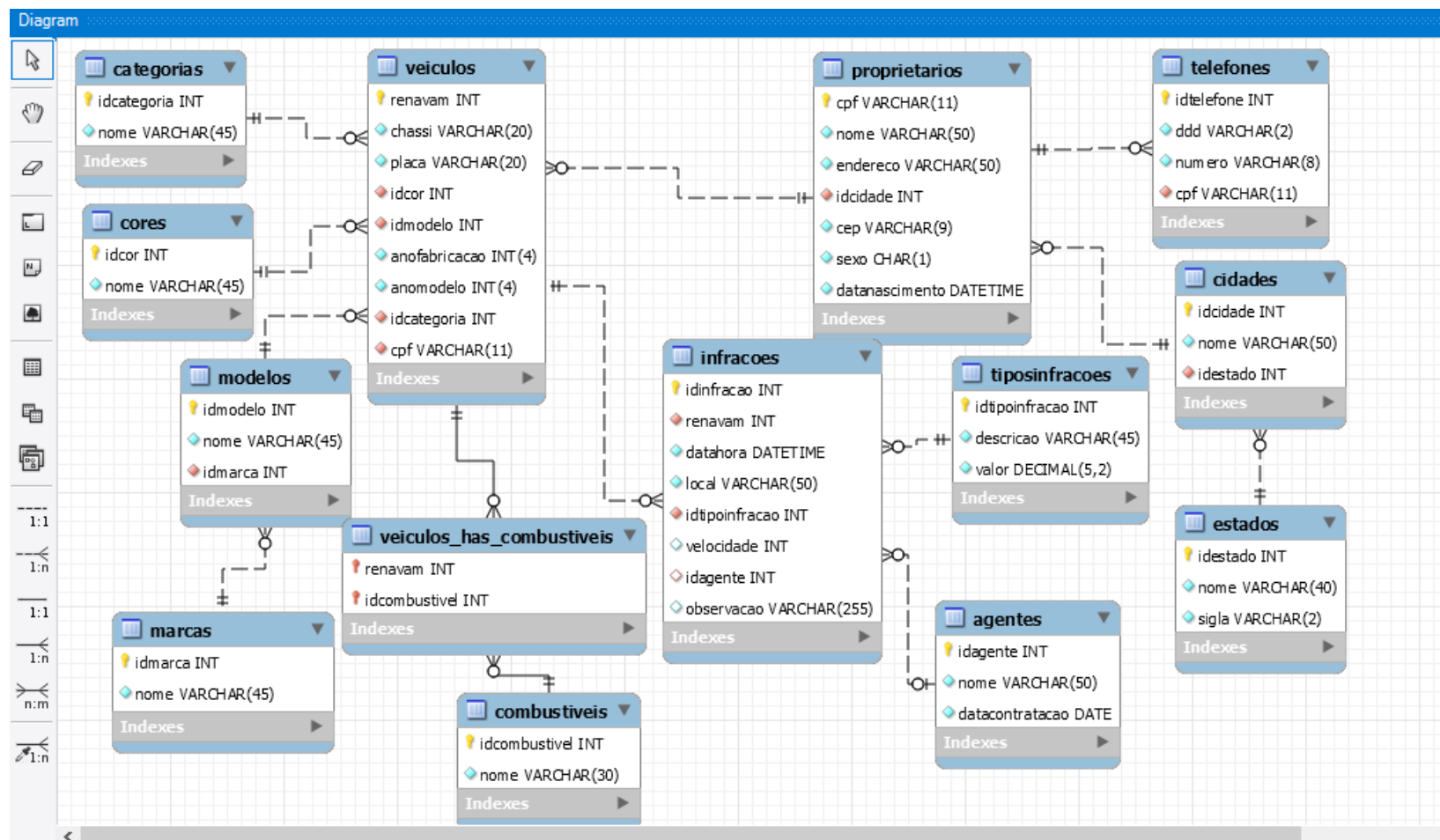
## Principais comandos SQL DML

DML (Data Manipulation Language)

- **SELECT**
- **INSERT**
- **DELETE**
- **UPDATE**
- COMMIT
- ROLLBACK

# AMBIENTE DE TRABALHO

Modelagem do veículos do Detran (fictício)



# SELECT BÁSICO

- **Consultar dados armazenados no BD**

```
SELECT atributo1, atributo2  
FROM tabela;
```

```
SELECT idmarca, nome  
FROM marcas;
```

```
SELECT *  
FROM tabela;
```

```
SELECT *  
FROM marcas;
```

# SELECT BÁSICO



Qual a diferença em performance do “select atributo1, atributo2 ...” para “select \*...”? Por que?



# INSERT

- **Adicionar uma linha na tabela existente (definindo os atributos)**

```
INSERT INTO tabela (atributo1, atributo2, ..., atributon)  
VALUES (valor1, valor2, ..., valorn);
```

```
INSERT INTO agentes (idagente, nome, datacontratacao)  
VALUES (10, "Leornado Brito", "2020-09-01");
```

Ou

```
INSERT INTO agentes (idagente, datacontratacao, nome)  
VALUES (10, "2020-09-01", "Leornado Brito");
```



# INSERT

- **Adicionar uma linha na tabela existente (sem declarar os atributos)**

INSERT INTO tabela

VALUES (valor1, valor2, ..., valorn);

INSERT INTO agentes

VALUES (10, "Leornado Brito", "2020-09-01");

# INSERT

- **Adicionar uma linha na tabela existente (campo com auto incremento)**

```
INSERT INTO tabela (atributo1, atributo2, ..., atributon)  
VALUES (null, valor2, ..., valorn);
```

```
INSERT INTO agentes (idagente, nome, datacontratacao)  
VALUES (null, "Leornado Brito", "2020-09-01");
```

```
INSERT INTO tabela (atributo2, ..., atributon)  
VALUES (valor2, ..., valorn);
```

```
INSERT INTO agentes (nome, datacontratacao,  
VALUES ("Leornado Brito", "2020-09-01");
```

# INSERT

- **Adicionar uma linha na tabela existente (campo com auto incremento)**

INSERT INTO tabela

VALUES (null, valor2, ..., valorn);

INSERT INTO agentes

VALUES (null, "Leornado Brito", "2020-09-01");

# INSERT



Todos os SGBD possuem auto incremento?

# UPDATE

- **Modificar valores de um ou mais atributos de uma ou mais linhas selecionadas**

UPDATE tabela

SET <atribuições>

WHERE <condição>;

UPDATE agentes

SET datacontratacao="2004-01-01"

WHERE nome="Francisco";

# UPDATE

```
set sql_safe_updates = 0;
```

Quando fazemos update usando condição que não é a chave primária.

# DELETE

- **Remover linhas na tabela existente**

DELETE FROM tabela WHERE <condição>;

DELETE FROM agentes  
WHERE idagente=6;



# DELETE



O que acontece quando é feito um delete e depois um insert numa tabela que a chave é um número auto incremento?

# DML



O que acontece se tentar incluir um registro sem colocar os campos obrigatórios? E no update?



O que acontece se tentar incluir um registro sem o valor válido de uma chave estrangeira? E no update?

# DML



O que acontece se tentar excluir um registro que é chave estrangeira em outra tabela e está em uso?



Qual a alternativa para isso nos sistemas?

# SELECT COM WHERE

- **Consultar dados armazenados no BD filtrando condições**

```
SELECT atributo1, atributo2  
FROM tabela  
WHERE <condição>;
```

```
SELECT idmarca, nome  
FROM marcas  
WHERE idmarca=2;
```

# SELECT COM WHERE

## Condições do comando WHERE

- Operações de comparação (=, >, <)
- Faixas (**BETWEEN** e **NOT BETWEEN**)
- Correspondência de caracteres (**LIKE** e **Not LIKE**)
- Valores desconhecidos (**IS NULL** e **IS NOT NULL**)
- Listas (**IN** e **NOT IN**)
- Combinações com **AND** e **OR**
- **NOT** pode negar qualquer expressão booleana e chaves como LIKE, NULL, BETWEEN e IN

# SELECT COM WHERE

## Operadores de comparação:

- = igual a
- > maior que
- < menor que
- >= maior que ou igual a
- <= menor que ou igual a
- != diferente
- <> diferente
- !> não maior que
- !< não menor que

# SELECT COM WHERE

Faixas BETWEEN e NOT BETWEEN :

**BETWEEN** – Opção usada para especificar uma faixa inclusiva; os valores extremos da faixa também são incluídos na busca.

**NOT BETWEEN** – Negação do BETWEEN. Exclui os valores especificados na faixa, inclusive os extremos.

```
SELECT idmarca, nome
```

```
FROM marcas
```

```
WHERE idmarca between 2 and 5;
```



# SELECT COM WHERE

## Correspondência de caracteres (LIKE e Not LIKE)

Seleciona linhas que contenham campos que correspondem a porções especificadas de uma série de caracteres (string de caracteres).

Utilizada apenas com dados do tipo char, varchar e datetime.

% – qualquer string com nenhum ou mais caracteres

\_ um único caracter

[ ] um único caracter na faixa especificada

[^] um único caracter fora da faixa especificada

Inclua os curingas e a string de caracteres entre aspas simples ou duplas.

```
SELECT idmarca, nome
```

```
FROM marcas
```

```
WHERE nome like "[L-P]";
```

Like não é =

# SELECT COM WHERE

## Listas (IN e NOT IN)

O operador de comparação IN permite ao usuário selecionar valores que correspondam a qualquer um de uma lista de valores.

```
SELECT idmarca, nome  
FROM marcas  
WHERE idmarca in (2,5,1,6);
```

```
SELECT idmarca, nome  
FROM marcas  
WHERE idmarca = 2 or idmarca =5 or idmarca=1 or idmarca= 6;
```

# ATIVIDADE EXTRACLASSE

- Lista de exercícios 1 de DML até o número 13.

# REFERÊNCIAS

## Básicas

- KORTH, H. F.; SILBERSCHATZ; S., SUDARSHAN. Sistema de Banco de Dados. Tradução da 6ª Ed. Elsevier, 2012.
- HEUSER, Carlos Alberto. Projeto de Banco de Dados. Porto Alegre, RS: Sagra Luzzato, 6ª edição, 2009.
- ELMASRI, R.; NAVATHE. Sistemas de Banco de Dados. Rio de Janeiro: Ed. Pearson, 7ª edição, 2018.

## Complementares

- DATE, C. J. Introdução a sistemas de banco de dados. RJ: Campus, 8ª edição, 2004.
- TEORREY, T. et al. Projeto e Modelagem de Banco de Dados. Ed. Elsevier, 2ª edição, 2014.
- MEDEIROS, L.F. Banco de Dados: princípios e prática. Ed. Intersaberes, 2013. ISBN: 9788582122181
- PUGA, S.; FRANÇA, E.; GOYA, M. Banco de dados: Implementação em SQL, PL/SQL e Oracle 11g. Ed. Pearson, 2013. ISBN: 9788581435329