Atividade 8 – DML

PÁGINA 9:

Criação das tabelas Medico, Ambulatorio, Pacientes e Consulta:

```sql
CREATE TABLE Medico (
    CRM INT PRIMARY KEY,
    nome VARCHAR(100),
    numeroAmbulatorio INT
);

CREATE TABLE Ambulatorio (
    numeroA INT PRIMARY KEY,
    andar INT,
    capacidade INT
);

CREATE TABLE Pacientes (
    idPaciente INT PRIMARY KEY,
    nome VARCHAR(100),
    dataNascimento DATE,
    endereco VARCHAR(200)
);

CREATE TABLE Consulta (
    idConsulta INT PRIMARY KEY,
    CRM INT,
    idPaciente INT,
    dataConsulta DATE,
    horaConsulta TIME,
    FOREIGN KEY (CRM) REFERENCES Medico(CRM),
    FOREIGN KEY (idPaciente) REFERENCES Pacientes(idPaciente)
);
```

Inserir 3 médicos na tabela:

```sql
INSERT INTO Medico VALUES(1, 'Dr. Carlos', 1);
INSERT INTO Medico VALUES(2, 'Dr. João', 2);
INSERT INTO Medico VALUES(3, 'Dra. Ana', 4);
```

Cadastrar 4 Ambulatórios:

```
INSERT INTO Ambulatorio VALUES (1, 1, 10);
INSERT INTO Ambulatorio VALUES (2, 2, 15);
INSERT INTO Ambulatorio VALUES (3, 1, 12);
INSERT INTO Ambulatorio VALUES (4, 3, 8);
```

Cadastrar 2 Pacientes:

```
INSERT INTO Pacientes VALUES (1, 'Felipe', '2004-07-02', 'Rua A, 123');
INSERT INTO Pacientes VALUES (2, 'Rafael', '2005-07-26', 'Rua B, 543');
```

Cadastrar 2 Consultas:

```
INSERT INTO Consulta VALUES (1, 1, 1, '2025-05-10', '10:00:00');
INSERT INTO Consulta VALUES (2, 2, 2, '2025-05-11', '11:30:00');
INSERT INTO Consulta VALUES (3, 3, 1, '2025-05-12', '14:00:00');
```

Alterar o número do Ambulatórios de todos os Médicos para 3:

```
SELECT * FROM Medico;

UPDATE Medico
SET numeroAmbulatorio = 3;

SELECT * FROM Medico;
```

Alterar o nome do Paciente 1 para ‘Pedro da Silva’:

```
SELECT * FROM Pacientes;

UPDATE Pacientes
SET nome = 'Pedro da Silva'
WHERE idPaciente = 1;

SELECT * FROM Pacientes;
```

DUMP da tabela Médico:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Medico](
    [CRM] [int] NOT NULL,
    [nome] [varchar](100) NULL,
    [numeroAmbulatorio] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Medico] ADD PRIMARY KEY CLUSTERED
(
    [CRM] ASC
)WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
GO
```

DUMP da tabela Ambulatório:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Ambulatorio](
    [numeroA] [int] NOT NULL,
    [andar] [int] NULL,
    [capacidade] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Ambulatorio] ADD PRIMARY KEY CLUSTERED
(
    [numeroA] ASC
)WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
GO
```

DUMP da tabela Pacientes:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Pacientes](
    [idPaciente] [int] NOT NULL,
    [nome] [varchar](100) NULL,
    [dataNascimento] [date] NULL,
    [endereco] [varchar](200) NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Pacientes] ADD PRIMARY KEY CLUSTERED
(
    [idPaciente] ASC
)WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
GO
```

DUMP da tabela Consultas:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Consulta](
    [idConsulta] [int] NOT NULL,
    [CRM] [int] NULL,
    [idPaciente] [int] NULL,
    [dataConsulta] [date] NULL,
    [horaConsulta] [time](7) NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Consulta] ADD PRIMARY KEY CLUSTERED
(
    [idConsulta] ASC
)WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Consulta]  WITH CHECK ADD FOREIGN KEY([CRM])
REFERENCES [dbo].[Medico] ([CRM])
GO
ALTER TABLE [dbo].[Consulta]  WITH CHECK ADD FOREIGN KEY([idPaciente])
REFERENCES [dbo].[Pacientes] ([idPaciente])
GO
```

PÁGINA 29 A 31:

Implementar o banco de dados e crias as expressões DDL:

```sql
CREATE TABLE Ambulatorio (
    numeroA INT PRIMARY KEY,
    andar INT,
    capacidade INT
);

CREATE TABLE Medico (
    CRM INT PRIMARY KEY,
    nome VARCHAR(100),
    idade INT,
    cidade VARCHAR(100),
    especialidade VARCHAR(100),
    numeroA INT,
    FOREIGN KEY (numeroA) REFERENCES Ambulatorio(numeroA)
);

CREATE TABLE Paciente (
    RG INT PRIMARY KEY,
    nome VARCHAR(100),
    idade INT,
    cidade VARCHAR(100),
    doenca VARCHAR(100)
);
```

```sql
CREATE TABLE Consulta (
    idConsulta INT PRIMARY KEY IDENTITY(1,1),
    CRM INT,
    RG INT,
    data DATE,
    hora TIME,
    FOREIGN KEY (CRM) REFERENCES Medico(CRM),
    FOREIGN KEY (RG) REFERENCES Paciente(RG)
);

CREATE TABLE Funcionario (
    RG INT PRIMARY KEY,
    nome VARCHAR(100),
    idade INT,
    cidade VARCHAR(100),
    salario DECIMAL(10, 2)
);
```

Expressões de visualização:

1.Buscar os dados dos pacientes que estão com dengue:

```sql
SELECT RG, nome, idade, cidade, doenca
FROM Paciente
WHERE doença = 'dengue';
```

2.Buscar os dados dos médicos cardiologistas com mais de 44 anos:

```sql
SELECT CRM, nome, idade, cidade, especialidade, numeroA
FROM Medico
WHERE especialidade = 'cardiologista' AND idade > 44;
```

3.Buscar os dados das Consultas, exceto aquelas marcadas para os médicos com CRM 4656 e 1879:

```sql
SELECT idConsulta, CRM, RG, data, hora
FROM Consulta
WHERE CRM NOT IN (4656, 1879);
```

4.Buscar os dados dos ambulatórios do quarto andar que ou tenham capacidade igual a 50 ou tenham número superior a 10:

```sql
SELECT numeroA, andar, capacidade
FROM Ambulatorio
WHERE andar = 4 AND (capacidade = 50 OR numeroA > 10);
```

5.Buscar o nome e a especialidade de todos os médicos:

```sql
SELECT nome, especialidade
FROM Medico
```

6.Buscar os números dos ambulatórios, exceto aqueles do segundo e quarto andares, que suportam mais de 50 pacientes:

```sql
SELECT numeroA
FROM Ambulatorio
WHERE andar NOT IN (2, 4) AND capacidade > 50;
```

7.Buscar os nomes dos médicos que tem consulta marcada e as datas das suas consultas:

```sql
SELECT M.nome, C.data
FROM Consulta C
JOIN Medico M ON C.CRM = M.CRM;
```

8.Buscar o número e a capacidade dos ambulatórios do quinto andar e o nome dos médicos que atendem neles:

```sql
SELECT A.numeroA, A.capacidade, M.nome
FROM Ambulatorio A
JOIN Medico M ON A.numeroA = M.numeroA
WHERE A.andar = 5;
```

9.Buscar o nome dos médicos e o nome dos seus pacientes com consulta marcada, assim como a data destas consultas:

```sql
SELECT M.nome AS nome_medico, P.nome AS nome_paciente, C.data
FROM Consulta C
JOIN Medico M ON C.CRM = M.CRM
JOIN Paciente P ON C.RG = P.RG;
```

10.Buscar os nomes dos médicos ortopedistas com consultas marcadas para o período da manhã (7hs – 12hs) do dia 20/06/24:

```sql
SELECT M.nome
FROM Consulta C
JOIN Medico M ON C.CRM = M.CRM
WHERE M.especialidade = 'Ortopedista'
  AND C.data = '2024-06-20'
  AND C.hora BETWEEN '07:00:00' AND '12:00:00';
```

11. buscar os nomes dos pacientes, com consultas marcadas para os médicos João Carlos Santos ou Maria Souza,
que estão com pneumonia:

```sql
SELECT P.nome
FROM Consulta C
JOIN Medico M ON C.CRM = M.CRM
JOIN Paciente P ON C.RG = P.RG
WHERE P.doenca = 'Pneumonia'
  AND M.nome IN ('João Carlos Santos', 'Maria Souza');
```

12.Buscar os nomes dos médicos e pacientes cadastrados no hospital:

```sql
SELECT nome
FROM Medico
UNION
SELECT nome
FROM Paciente;
```

13.Buscar os nomes e idade dos médicos, pacientes e funcionários que residem em Ribeirão das Neves:

```sql
SELECT nome, idade FROM Medico WHERE cidade = 'Ribeirão das Neves'
UNION
SELECT nome, idade FROM Paciente WHERE cidade = 'Ribeirão das Neves'
UNION
SELECT nome, idade FROM Funcionario WHERE cidade = 'Ribeirão das Neves';
```

14.Buscar os nomes e RGs dos funcionários que recebem salários abaixo de R$ 3000,00 e que não estão internados como pacientes:

```sql
SELECT F.nome, F.RG
FROM Funcionario F
WHERE F.salario < 3000
  AND F.RG NOT IN (SELECT RG FROM Paciente);
```

15.Buscar os números dos ambulatórios onde nenhum médico dá atendimento:

```sql
SELECT A.numeroA
FROM Ambulatorio A
WHERE A.numeroA NOT IN (
    SELECT DISTINCT numeroA FROM Medico
);
```

16.Buscar os nomes e RGs dos funcionários que estão internados como pacientes:

```sql
SELECT F.nome, F.RG
FROM Funcionario F
JOIN Paciente P ON F.RG = P.RG;
```

Expressões em Álgebra Relacional:

1.Buscar os dados dos pacientes que estão com dengue:

$$\sigma(\text{doença} = \text{'Dengue'})(\text{Paciente})$$

2.Buscar os dados dos médicos cardiologistas com mais de 44 anos:

$$\sigma(\text{especialidade} = \text{'Cardiologia'} \land \text{idade} > 44)(\text{Medico})$$

3.Buscar os dados das Consultas, exceto aquelas marcadas para os médicos com CRM 4656 e 1879:

$$\sigma(\text{CRM} \neq 4656 \land \text{CRM} \neq 1879)(\text{Consulta})$$

4.Buscar os dados dos ambulatórios do quarto andar que ou tenham capacidade igual a 50 ou tenham número superior a 10:

$$\sigma(\text{andAR} = 4 \land (\text{capacidade} = 50 \lor \text{numero} > 10))(\text{Ambulatorio})$$

5.Buscar o nome e a especialidade de todos os médicos:

$$\pi(\text{nome, especialidade})(\text{Medico})$$

6.Buscar os números dos ambulatórios, exceto aqueles do segundo e quarto andares, que suportam mais de 50 pacientes:

$$\pi(\text{numero})(\sigma(\text{andAR} \neq 2 \land \text{andAR} \neq 4 \land \text{capacidade} > 50)(\text{Ambulatorio}))$$

7.Buscar os nomes dos médicos que tem consulta marcada e as datas das suas consultas:

$$\pi(\text{Medico.nome, Consulta.data})(\text{Medico} \bowtie \text{Medico.CRM} = \text{Consulta.CRM})$$

8.Buscar o número e a capacidade dos ambulatórios do quinto andar e o nome dos médicos que atendem neles:

$$\pi(\text{Ambulatorio.numero, Ambulatorio.capacidade, Medico.nome})((\sigma(\text{andAR} = 5)(\text{Ambulatorio})) \bowtie \text{Ambulatorio.numero} = \text{Medico.numeroAmb})$$

9.Buscar o nome dos médicos e o nome dos seus pacientes com consulta marcada, assim como a data destas consultas:

$$\pi(\text{Medico.nome, Paciente.nome, Consulta.data})((\text{Consulta} \bowtie \text{Consulta.CRM} = \text{Medico.CRM}) \bowtie \text{Consulta.idPaciente} = \text{Paciente.id})$$

10.Buscar os nomes dos médicos ortopedistas com consultas marcadas para o período da manhã (7hs – 12hs) do dia 20/06/24:

$$\pi(\text{Medico.nome})(\sigma(\text{especialidade} = \text{'Ortopedia'} \wedge \text{data} = \text{'2024-06-20'} \wedge \text{hora} \geq \text{'07:00'} \wedge \text{hora} \leq \text{'12:00'})(\text{Consulta} \bowtie \text{Medico}))$$

11. buscar os nomes dos pacientes, com consultas marcadas para os médicos João Carlos Santos ou Maria Souza,
que estão com pneumonia:

$$\pi(\text{Paciente.nome})($$

σ(doença = 'Pneumonia' ∧ (Medico.nome = 'João Carlos Santos' ∨
Medico.nome = 'Maria Souza'))(
(Consulta ⋈ Medico) ⋈ Consulta.idPaciente = Paciente.id))

12.Buscar os nomes dos médicos e pacientes cadastrados no hospital:

π(nome)(Medico) ∪ π(nome)(Paciente)

13.Buscar os nomes e idade dos médicos, pacientes e funcionários que residem em Ribeirão das Neves:

π(nome, idade)(σ(cidade = 'Ribeirão das Neves')(Medico)) ∪
π(nome, idade)(σ(cidade = 'Ribeirão das Neves')(Paciente)) ∪
π(nome, idade)(σ(cidade = 'Ribeirão das Neves')(Funcionario))

14.Buscar os nomes e RGs dos funcionários que recebem salários abaixo de R$ 3000,00 e que não estão internados como pacientes:

π(nome, RG)(σ(salario < 3000)(Funcionario)) −
π(nome, RG)(Funcionario ⋈ Internado)

15.Buscar os números dos ambulatórios onde nenhum médico dá atendimento:

π(numero)(Ambulatorio) − π(numeroAmb)(Medico)

16.Buscar os nomes e RGs dos funcionários que estão internados como pacientes:

π(Funcionario.nome, Funcionario.RG)(
Funcionario ⋈ Funcionario.id = Paciente.idFuncionario)

DUMP da tabela Ambulatorio:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Ambulatorio](
    [numeroA] [int] NOT NULL,
    [andar] [int] NULL,
    [capacidade] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Ambulatorio] ADD PRIMARY KEY CLUSTERED
(
    [numeroA] ASC
)WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
GO
```

DUMP da tabela Consulta:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Consulta](
    [idConsulta] [int] IDENTITY(1,1) NOT NULL,
    [CRM] [int] NULL,
    [RG] [int] NULL,
    [data] [date] NULL,
    [hora] [time](7) NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Consulta] ADD PRIMARY KEY CLUSTERED
(
    [idConsulta] ASC
)WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Consulta]  WITH CHECK ADD FOREIGN KEY([CRM])
REFERENCES [dbo].[Medico] ([CRM])
GO
ALTER TABLE [dbo].[Consulta]  WITH CHECK ADD FOREIGN KEY([RG])
REFERENCES [dbo].[Paciente] ([RG])
GO
```

DUMP da tabela Funcionario:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Funcionario](
    [RG] [int] NOT NULL,
    [nome] [varchar](100) NULL,
    [idade] [int] NULL,
    [cidade] [varchar](100) NULL,
    [salario] [decimal](10, 2) NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Funcionario] ADD PRIMARY KEY CLUSTERED
(
    [RG] ASC
)WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
GO
```

DUMP da tabela Medico:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Medico](
    [CRM] [int] NOT NULL,
    [nome] [varchar](100) NULL,
    [idade] [int] NULL,
    [cidade] [varchar](100) NULL,
    [especialidade] [varchar](100) NULL,
    [numeroA] [int] NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Medico] ADD PRIMARY KEY CLUSTERED
(
    [CRM] ASC
)WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Medico]  WITH CHECK ADD FOREIGN KEY([numeroA])
REFERENCES [dbo].[Ambulatorio] ([numeroA])
GO
```

DUMP da tabela Paciente:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Paciente](
    [RG] [int] NOT NULL,
    [nome] [varchar](100) NULL,
    [idade] [int] NULL,
    [cidade] [varchar](100) NULL,
    [doenca] [varchar](100) NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Paciente] ADD PRIMARY KEY CLUSTERED
(
    [RG] ASC
)WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
GO
```