



Formativa 2

1. Apresente o resultado obtido para cada uma das consultas a seguir, no formato de tabela, e explique o que o comando faz:

```
-- a)

SELECT TC.NomeCurso, TA.NomeAluno
FROM tabaluno AS TA
INNER JOIN tabcurso AS TC ON (TA.fk_CodCurso = TC.CodCurso)
ORDER BY TC.NomeCurso, TA.NomeAluno;
```

NomeCurso	NomeAluno
Técnica em Administração	Antônia Mendes
Técnica em Administração	Arthur Fontes
Técnica em Eletrônica	Carlos Torres
Técnica em Eletrônica	José Silva
Técnica em Informática	Fernando Santos
Técnica em Informática	Maria Flores

- Nomeia os cursos e seus respectivos alunos em ordem alfabética.

```
-- b)

SELECT TC.NomeCurso, COUNT(*) AS TotalAlunos
FROM tabaluno AS TA
INNER JOIN tabcurso AS TC ON (TA.fk_CodCurso = TC.CodCurso)
GROUP BY TC.NomeCurso
ORDER BY TC.NomeCurso;
```

NomeCurso	TotalAlunos
Técnica em Administração	2
Técnica em Eletrônica	2
Técnica em Informática	2

- Conta a quantidade de alunos em cada curso.

```
-- c)
SELECT TC.NomeCurso, SUM(TAD.NumeroFaltas) AS TotalFaltasAlunos
FROM tabaluno AS TA
INNER JOIN tabcurso AS TC ON (TA.fk_CodCurso = TC.CodCurso)
INNER JOIN tabAlunoDisciplina AS TAD ON (TAD.fk_CodMatricula = TA.CodMatricula)
GROUP BY TC.NomeCurso
ORDER BY TC.NomeCurso;
```

NomeCurso	TotalFaltasAlunos
Técnica em Administração	30
Técnica em Eletrônica	30
Técnica em Informática	30

- Conta a quantidade de faltas de Maria, José e Arthur em seus cursos.

```
-- d)
SELECT TC.NomeCurso, SUM(TAD.NumeroFaltas) AS TotalFaltasAlunos2020
FROM tabaluno AS TA
INNER JOIN tabcurso AS TC ON (TA.fk_CodCurso = TC.CodCurso)
INNER JOIN tabAlunoDisciplina AS TAD ON (TAD.fk_CodMatricula = TA.CodMatricula)
WHERE TAD.Ano = 2020
GROUP BY TC.NomeCurso
ORDER BY TC.NomeCurso;
```

NomeCurso	TotalFaltasAlunos2020
Técnica em Administração	10
Técnica em Eletrônica	10

NomeCurso	TotalFaltasAlunos2020
Técnica em Informática	20

- Conta a quantidade de faltas de Maria, José e Arthur no ano de 2020.

```
-- e)
SELECT TD.NomeDisciplina AS Disciplina, COUNT(*) AS TotalAlunos
FROM tabDisciplina AS TD
INNER JOIN tabAlunoDisciplina AS TAD ON (TD.CodDisciplina = TAD.fk_CodDisciplina)
GROUP BY TD.NomeDisciplina
ORDER BY TD.NomeDisciplina;
```

Disciplina	TotalAlunos
Biologia	1
Circuitos Integrados	1
Língua Portuguesa	2
Lógica de Programação	1
Matemática	1

- Conta o total de alunos em cada disciplina.

```
-- f)
SELECT TD.NomeDisciplina AS Disciplina, COUNT(*) AS TotalAlunos,
SUM(NumeroFaltas) AS TotalFaltas
FROM tabDisciplina AS TD
INNER JOIN tabAlunoDisciplina AS TAD ON (TD.CodDisciplina = TAD.fk_CodDisciplina)
GROUP BY TD.NomeDisciplina
ORDER BY TD.NomeDisciplina;
```

Disciplina	TotalAlunos	TotalFaltas
Biologia	1	10
Circuitos Integrados	1	10
Língua Portuguesa	2	30
Lógica de Programação	1	20
Matemática	1	20

- Mostra o total de alunos em cada disciplina e as faltas de cada um.

```
-- g)
SELECT TD.NomeDisciplina AS Disciplina, COUNT(*) AS TotalAlunos,
AVG(TAD.NotaMedia) AS NotaMédia
FROM tabDisciplina AS TD
INNER JOIN tabAlunoDisciplina AS TAD ON (TD.CodDisciplina = TAD.fk_CodDisciplina)
GROUP BY TD.NomeDisciplina
ORDER BY TD.NomeDisciplina;
```

Disciplina	TotalAlunos	NotaMedia
Biologia	1	10
Circuitos Integrados	1	10
Língua Portuguesa	2	7.5
Lógica de Programação	1	5
Matemática	1	10

- Mostra o total de alunos em cada disciplina e a nota média de cada um.

```
-- h)
SELECT TC.NomeCurso, TD.NomeDisciplina, TA.NomeAluno, TAD.NotaMedia
FROM tabdisciplina AS TD
INNER JOIN tabalunodisciplina AS TAD ON (TD.CodDisciplina = TAD.fk_CodDisciplina)
INNER JOIN tabaluno AS TA ON (TA.CodMatricula = TAD.fk_CodMatricula)
INNER JOIN tabcurso AS TC ON (TA.fk_CodCurso = TC.CodCurso)
ORDER BY NomeCurso, NomeDisciplina, NomeAluno;
```

NomeCurso	NomeDisciplina	NomeAluno	NotaMedia
Técnica em Administração	Biologia	Arthur Fontes	10
Técnica em Administração	Língua Portuguesa	Arthur Fontes	5
Técnica em Eletrônica	Circuitos Integrados	José Silva	10
Técnica em Eletrônica	Matemática	José Silva	10
Técnica em Informação	Língua Portuguesa	Maria Flores	10
Técnica em Informação	Lógica de Programação	Maria Flores	5

- Mostra as disciplinas cursadas por Maria, José e Arthur dentro de seus cursos e suas médias em cada uma.

```
-- i)
SELECT TC.NomeCurso, TD.NomeDisciplina, TA.NomeAluno, TAD.NotaMedia
FROM tabdisciplina AS TD
INNER JOIN tabalunodisciplina AS TAD ON (TD.CodDisciplina = TAD.fk_CodDisciplina)
INNER JOIN tabaluno AS TA ON (TA.CodMatricula = TAD.fk_CodMatricula)
INNER JOIN tabcurso AS TC ON (TA.fk_CodCurso = TC.CodCurso)
WHERE (TC.NomeCurso = 'Técnico em Informática' OR
TC.NomeCurso = 'Técnico em Administração')
ORDER BY NomeDisciplina, NomeAluno;
```

NomeCurso	NomeDisciplina	NomeAluno	NotaMedia
Técnica em Administração	Biologia	Arthur Fontes	10
Técnica em Administração	Língua Portuguesa	Arthur Fontes	5

NomeCurso	NomeDisciplina	NomeAluno	NotaMedia
Técnica em Informação	Língua Portuguesa	Maria Flores	10
Técnica em Informação	Lógica de Programação	Maria Flores	5

- Mostra as disciplinas cursadas por Maria e Arthur dentro de seus cursos (correspondentes a Técnica em Informação ou Técnica em Administração) e suas médias em cada uma.

```
-- j)
DROP PROCEDURE IF EXISTS alunosCurso;
DELIMITER $$
CREATE PROCEDURE alunosCurso (IN nomeCurso VARCHAR(50), OUT notaMedia FLOAT)
BEGIN
    -- Resultado do AVG é atribuído à notaMedia = parâmetro de OUTPUT
    SELECT AVG(TAD.NotaMedia) INTO notaMedia
    FROM tabdisciplina AS TD
    INNER JOIN tabalunodisciplina AS TAD ON (TD.CodDisciplina = TAD.fk_CodDisciplina)
    INNER JOIN tabaluno AS TA ON (TA.CodMatricula = TAD.fk_CodMatricula)
    INNER JOIN tabcurso AS TC ON (TA.fk_CodCurso = TC.CodCurso)
    WHERE TC.NomeCurso LIKE nomeCurso ;
    -- Ao ser executada, a SP exibe o resultado do SELECT
    SELECT TC.NomeCurso, TD.NomeDisciplina, TA.NomeAluno, TAD.NotaMedia,
    TAD.NumeroFaltas
    FROM tabdisciplina AS TD
    INNER JOIN tabalunodisciplina AS TAD ON (TD.CodDisciplina = TAD.fk_CodDisciplina)
    INNER JOIN tabaluno AS TA ON (TA.CodMatricula = TAD.fk_CodMatricula)
    INNER JOIN tabcurso AS TC ON (TA.fk_CodCurso = TC.CodCurso)
    WHERE TC.NomeCurso LIKE nomeCurso -- TC.NomeCurso PARECIDO com nomeCurso
    ORDER BY NomeDisciplina, NomeAluno;
END $$
DELIMITER ;

-- 1.
SET @notaMediaCurso = 0.0; -- Declara e atribui variável de sessão
SET @nCurso = 'Técnico em Informática'; -- Declara e atribui variável de sessão
CALL alunosCurso (@nCurso, @notaMediaCurso);
SELECT @nCurso AS Curso, @notaMediaCurso AS NotaMediaCurso;

-- 2.
SET @nCurso = 'Técnico em Eletrônica'; -- Atribui variável de sessão
CALL alunosCurso (@nCurso, @notaMediaCurso);
SELECT @nCurso AS Curso, @notaMediaCurso AS NotaMediaCurso;
```

```
-- 3.
SET @nCurso = 'Técnico em Administração'; -- Atribui variável de sessão
CALL alunosCurso (@nCurso, @notaMediaCurso);
SELECT @nCurso AS Curso, @notaMediaCurso AS NotaMediaCurso;
```

1. Mostra a média do curso 'Técnica em Informação'.

Curso	NotaMediaCurso
Técnica em Informação	7.5

2. Mostra a média do curso 'Técnica em Eletrônica'.

Curso	NotaMediaCurso
Técnica em Eletrônica	10

3. Mostra a média do curso 'Técnica em Administração'.

Curso	NotaMediaCurso
Técnica em Administração	7.5

2. a) Explique o que é a 3FN para normalização, e como utilizar esse recurso?

A 3FN (Terceira Forma Normal) serve para evitar redundância e garantir que cada atributo em uma tabela dependa apenas da chave primária, sem dependências transitivas

Por exemplo, se a cidade de um cliente depende do ID do cliente, que por sua vez depende da chave primária, isso é uma dependência transitiva que deve ser evitada através da normalização.

b) Explique o que é um Trigger SQL e quando ele deve ser utilizado?

Trigger é um gatilho que provocará algum acontecimento caso ele seja ativado.

Por exemplo, ter um trigger chamado "After INSERT on EXP". Todas as vezes que um "INSERT into EXP" for realizado, esse trigger será ativado, realizando a

função atribuída à ele.

Prova Formativa de Banco de Dados (2)

(1) a) Nome Curso Nome Aluno

Técnico em Administração	Antônio Mendes
Técnico em Administração	Arthur Fontes
Técnico em Eletrônica	Carlos Torres
Técnico em Eletrônica	João Silva
Técnico em Informática	Fernando Gomes
Técnico em Informática	Maria Feres

• Nomeia os cursos e seus respectivos alunos em ordem alfabética

b) Nome Curso Total Alunos

Técnico em Administração	2
Técnico em Eletrônica	2
Técnico em Informática	2

• Conta a quantidade de alunos em cada curso

c) Nome Curso Total Faltas Alunos

Técnico em Administração	30
Técnico em Eletrônica	30
Técnico em Informática	30

• Conta o total de faltas de Maria, João e Arthur em seus cursos

d) Nome Curso Total Faltas Alunos 2020

Técnico em Administração	10
Técnico em Eletrônica	10
Técnico em Informática	20

• Conta o total de faltas de Maria, João e Arthur no ano de 2020.

e) Disciplina Total Alunos

Biologia	1
Circuitos Integrados	1
Lógica Portlógica	2
Lógica de Programação	1
Matemática	1

• Conta o total de alunos em cada matéria

1) Disciplina	Total Alunos	Total Faltas	
Biologia	1	10	
Circuitos Integrados	1	10	• mostra o total de
Lingua Portuguesa	2	30	alunos em cada dis-
Logica de Programação	1	20	c. as faltas de cada
Matemática	1	20	um.

2) Disciplina	Total Alunos	Nota Média	
Biologia	1	10	
Circuitos Integrados	1	10	• mostra o total de
Lingua Portuguesa	2	7.5	alunos em cada disci-
Logica de Programação	1	5	e a nota média
Matemática	1	10	de cada um.

3) Nome Curso	Nome Disciplina	Nome Aluno	Nota Média
Tec. em Adm.	Biologia	Arthur Fontes	10
Tec. em Adm.	Lingua Portuguesa	Arthur Fontes	5
Tec. em Elec.	Circuitos Integrados	José Silva	10
Tec. em Elec.	Matemática	José Silva	10
Tec. em Info.	Lingua Portuguesa	Maria Flores	10
Tec. em Info.	Logica de Programação	Maria Flores	5

4) Nome Curso	Nome Disciplina	Nome Aluno	Nota Média
Tec. em Adm.	Biologia	Arthur Fontes	10
Tec. em Adm.	Lingua Portuguesa	Arthur Fontes	5
Tec. em Info.	Lingua Portuguesa	Maria Flores	10
Tec. em Info.	Logica de Programação	Maria Flores	5

5) 1. Curso	Nota Média Curso	
Tec. em Info.	7.5	• nota média por curso "Tecnico em
		Informática"

6) 2. Curso	Nota Média Curso	
Tec. em Elec.	10	• nota média em "Tecnico em Eletrônica"

7) 3. Curso	Nota Média Curso	
Tec. em Adm.	7.5	• nota média em "Tecnico em Administração"

② a) 1. 3FN (Terceira Normal) serve para evitar redundância e garantir que cada atributo em uma tabela dependa apenas da chave primária, sem dependências transitivas.

Por exemplo, o endereço de um cliente depende do ID do cliente, que por sua vez, depende da chave primária. Isso é uma dependência transitiva que deve ser evitada através da normalização.

b) Trigger é um programa que provocará algum acontecimento, caso ele seja ativado.

Por exemplo, tem um trigger "After INSERT on Exp". Todas as vezes que um "INSERT into Exp" esse trigger será ativado, executando a função atribuída a ele.