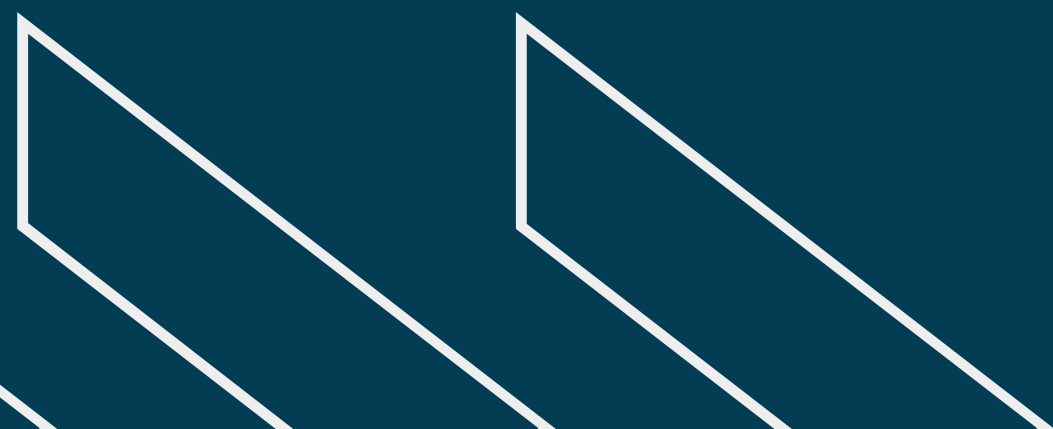


Apresentação
**MODERNIZAÇÃO DO
ARMAZENAMENTO E
GERENCIAMENTO DE
DADOS DA RESILIA**



SQUAD #1

Ysmael Marques

Nilton de Maria

Diego Carvalho Feitosa

Leonardo Xavier

Larissa Abrahão



PESSOA
COFACILITADORA



PESSOA
GESTORA DE
CONHECIMENTO



PESSOA GESTORA
DE GENTE E
ENGAJAMENTO



PESSOA
COLABORADORA I



PESSOA
COLABORADORA II

CONTEXTO

Você e a sua equipe foram escalados pela Resilia para modernizar o processo de armazenamento de dados e construção para gerenciamento da estrutura de ensino da empresa.

Para isso, vocês devem se atentar para o descritivo que será apresentado :

Hoje dentro da Resilia, são armazenadas diversas informações do braço de ensino da empresa como dados sobre os alunos, facilitadores, turmas, módulos e cursos em planilhas. Essas informações são colocadas em planilhas diferentes, dificultando muitas das vezes a extração de dados estratégicos para a empresa.

O QUE É PARA FAZER?

1. Gerar uma representação das entidades e seus respectivos atributos e relacionamentos;

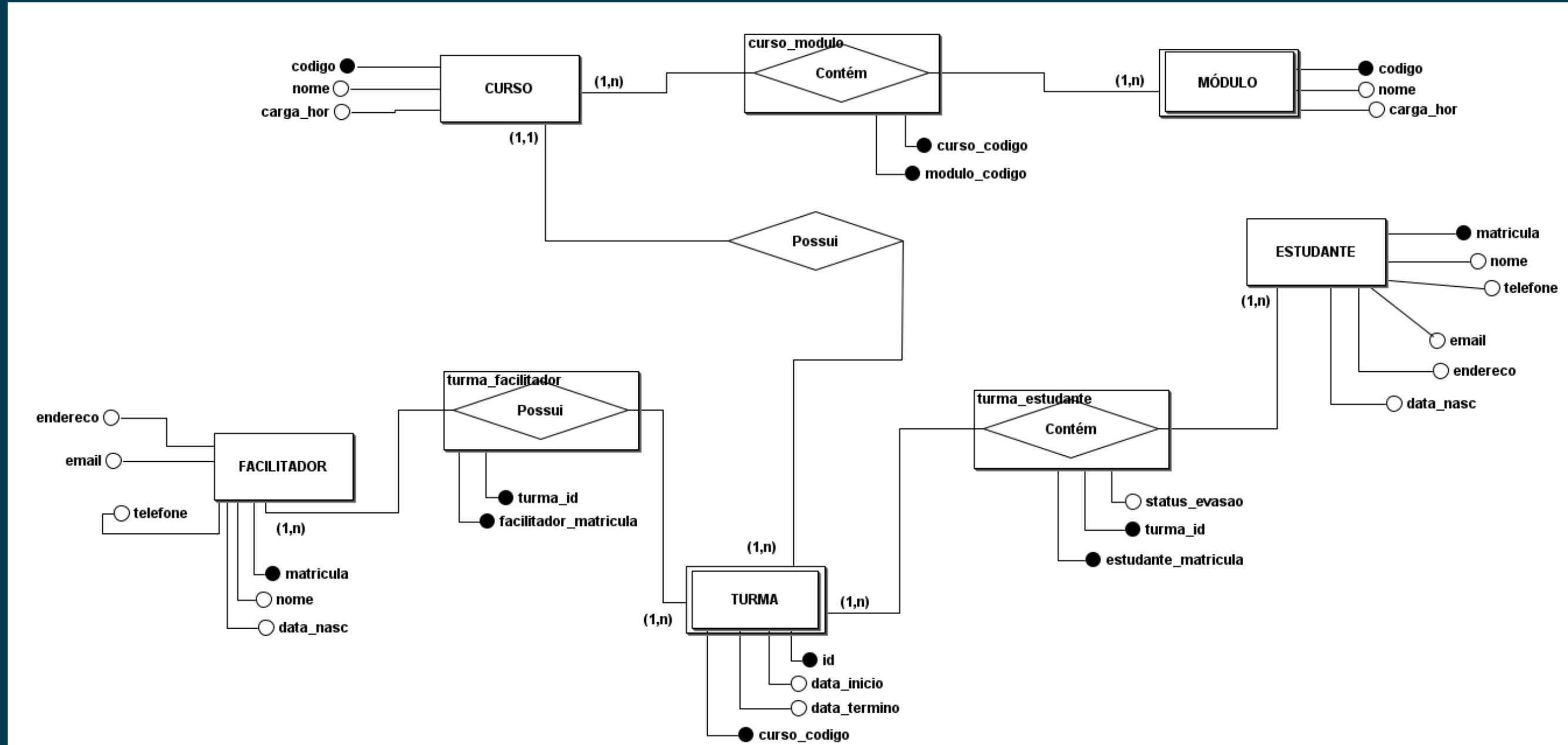
2. Criar a modelagem do banco de dados.

3. Criar os scripts SQL para criação do banco de dados e das tabelas com seus respectivos atributos;

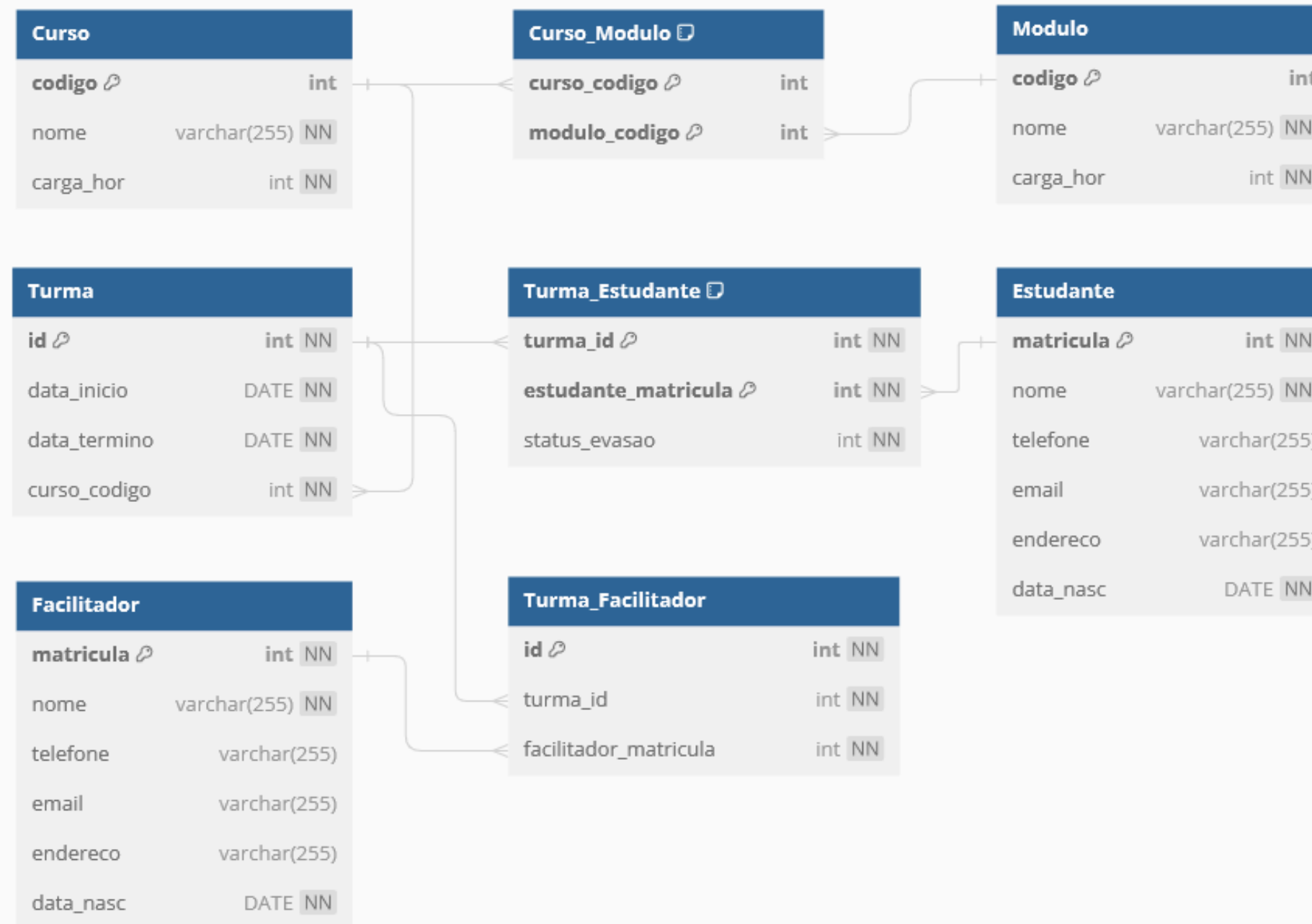
4. Criar scripts SQL para inserção dos dados nas tabelas e Executar consultas para gerar informações estratégicas para a área de ensino da Resilia



Modelo Conceitual:



Modelo Lógico:



Criação do Banco de Dados:

```
1 CREATE TABLE `Modulo` (  
2     `codigo` int PRIMARY KEY,  
3     `nome` varchar(255) NOT NULL,  
4     `carga_hor` int NOT NULL  
5 );  
6  
7 CREATE TABLE `Curso` (  
8     `codigo` int PRIMARY KEY,  
9     `nome` varchar(255) NOT NULL,  
10    `carga_hor` int NOT NULL  
11 );  
12  
13 CREATE TABLE `Curso_Modulo` (  
14     `curso_codigo` int,  
15     `modulo_codigo` int,  
16     PRIMARY KEY (`curso_codigo`, `modulo_codigo`),  
17     FOREIGN KEY (`curso_codigo`) REFERENCES `Curso` (`codigo`),  
18     FOREIGN KEY (`modulo_codigo`) REFERENCES `Modulo` (`codigo`)  
19 );  
20  
21 CREATE TABLE `Facilitador` (  
22     `matricula` int UNIQUE PRIMARY KEY NOT NULL,  
23     `nome` varchar(255) NOT NULL,  
24     `telefone` varchar(255),  
25 );
```

#	Time	Action	Message
3	08:46:38	use newdb	0 row(s) affected
4	08:46:53	CREATE TABLE `Modulo` (`codigo` int PRIMARY KEY, `nome` varchar(255) NOT NULL, `c...	0 row(s) affected
5	08:46:53	CREATE TABLE `Curso` (`codigo` int PRIMARY KEY, `nome` varchar(255) NOT NULL, `ca...	0 row(s) affected
6	08:46:53	CREATE TABLE `Curso_Modulo` (`curso_codigo` int, `modulo_codigo` int, PRIMARY KEY (...	0 row(s) affected
7	08:46:53	CREATE TABLE `Facilitador` (`matricula` int UNIQUE PRIMARY KEY NOT NULL, `nome` va...	0 row(s) affected
8	08:46:53	CREATE TABLE `Tuma` (`id` int UNIQUE PRIMARY KEY NOT NULL AUTO_INCREMENT, ...	0 row(s) affected

Inserção de Dados:

```
1 • INSERT INTO `Modulo` (`codigo`, `nome`, `carga_hor`)
2 VALUES
3 (1, 'Introdução à Programação', 40),
4 (2, 'Banco de Dados', 30),
5 (3, 'Desenvolvimento Web', 50),
6 (4, 'Machine Learning', 60),
7 (5, 'Redes de Computadores', 45);
8
9 • INSERT INTO `Curso` (`codigo`, `nome`, `carga_hor`)
10 VALUES
11 (1, 'Ciência da Computação', 240),
12 (2, 'Engenharia de Software', 220),
13 (3, 'Sistemas de Informação', 200),
14 (4, 'Inteligência Artificial', 180),
15 (5, 'Segurança da Informação', 210);
16
17 • INSERT INTO `Facilitador` (`matricula`, `nome`, `telefone`, `email`, `endereco`, `data_nasc`)
18 VALUES
19 (1001, 'João Silva', '123-456-789', 'joao.silva@example.com', 'Rua A, 123', '1980-05-15'),
20 (1002, 'Maria Santos', '987-654-321', 'maria.santos@example.com', 'Rua B, 456', '1975-10-20')
```





✓	19	08:46:59	INSERT INTO `Turma_Estudante` (`turma_id`, `estudante_matricula`, `status_evasao`) VALUES...	35 row(s) affected Records: 35
✓	20	08:47:10	CREATE TRIGGER `log_status_evasao` AFTER UPDATE ON `Turma_Estudante` FOR EACH ...	0 row(s) affected
✓	21	08:47:24	UPDATE `Turma_Estudante` SET `status_evasao` = 1 WHERE `estudante_matricula` = 2002	0 row(s) affected Rows matched
✓	22	08:47:30	SELECT * FROM `Log_Status_Evasao` LIMIT 0, 1000	0 row(s) returned

Primeira Questão:

Quantidade total de estudantes cadastrados no banco.

```
1 • SELECT COUNT(*) AS total_estudantes FROM Estudante;
```

<

Result Grid   Filter Rows: | Export:  | Wrap Cell Content: 





	total_estudantes
▶	35

Segunda Questão:

Quais pessoas facilitadoras atuam em mais de uma turma.

```
1 • SELECT
2     f.matricula,
3     f.nome,
4     COUNT(tf.turma_id) AS total_turmas
5 FROM
6     Facilitador f
7 INNER JOIN
8     Turma_Facilitador tf ON f.matricula = tf.facilitador_matricula
9 GROUP BY
10    f.matricula, f.nome
11 HAVING
12    COUNT(tf.turma_id) > 1;
13
```

<

Result Grid   Filter Rows: | Export:  | Wrap Cell Content: 

	matricula	nome	total_turmas
▶	1002	Maria Santos	2
	1003	Carlos Oliveira	2

Terceira Questão:

Porcentagem de estudantes com status de evasão agrupados por turma.

```
CREATE VIEW Porcentagem_Evasao_Turma AS
SELECT
  TE.turma_id,
  COUNT(TE.estudante_matricula) AS total_estudantes,
  SUM(CASE WHEN TE.status_evasao = 1 THEN 1 ELSE 0 END) AS evadidos,
  CONCAT(FORMAT((SUM(CASE WHEN TE.status_evasao = 1 THEN 1 ELSE 0 END) / COUNT(TE.estudante_matricula)) * 100, 2), '%') AS percentual_evasao
FROM
  Turma_Estudante TE
GROUP BY
  TE.turma_id;
```

Time	Action	Message
08:49:54	SELECT * FROM newdb.log_status_evasao LIMIT 0, 1000	0 row(s) returned
08:50:30	UPDATE 'Turma_Estudante' SET 'status_evasao' = 0 WHERE 'estudante_matricula' = 2002	1 row(s) affected
08:50:38	SELECT * FROM 'Log_Status_Evasao' LIMIT 0, 1000	1 row(s) returned
09:03:04	SELECT COUNT(*) AS total_estudantes FROM Estudante LIMIT 0, 1000	1 row(s) returned
09:05:16	SELECT f.matricula, f.nome, COUNT(f.turma_id) AS total_turmas FROM Facilitador f INNER JOIN Turma f2 ON f.turma_id = f2.turma_id	2 row(s) returned
09:09:22	CREATE VIEW Porcentagem_Evasao_Turma AS SELECT TE.turma_id, COUNT(TE.estudante_matricula) AS total_estudantes, SUM(CASE WHEN TE.status_evasao = 1 THEN 1 ELSE 0 END) AS evadidos, CONCAT(FORMAT((SUM(CASE WHEN TE.status_evasao = 1 THEN 1 ELSE 0 END) / COUNT(TE.estudante_matricula)) * 100, 2), '%') AS percentual_evasao FROM Turma_Estudante TE GROUP BY TE.turma_id;	0 row(s) affected

Criação

```
1 • SELECT * FROM Porcentagem_Evasao_Turma;
```

	turma_id	total_estudantes	evadidos	percentual_evasao
▶	1	15	5	33.33%
	2	10	4	40.00%
	3	10	4	40.00%

Inserção

Quarta Questão:

Crie um trigger para ser disparado quando o atributo status de um estudante for atualizado e inserir um novo dado em uma tabela de log.

```
1  DELIMITER $$
2  CREATE TRIGGER `log_status_evasao`
3  AFTER UPDATE ON `Turma_Estudante`
4  FOR EACH ROW
5  BEGIN
6      IF OLD.status_evasao <> NEW.status_evasao THEN
7          INSERT INTO `Log_Status_Evasao` (`estudante_matricula`, `data_modificacao`, `status_an`
8          VALUES (NEW.estudante_matricula, NOW(), OLD.status_evasao, NEW.status_evasao);
9      END IF;
10 END$$
11 DELIMITER ;
12
```

✓	19	08:46:59	INSERT INTO `Turma_Estudante` (`turma_id`, `estudante_matricula`, `status_evasao`) VALUES...	35 row(s) affected Records: 35
✓	20	08:47:10	CREATE TRIGGER `log_status_evasao` AFTER UPDATE ON `Turma_Estudante` FOR EACH ...	0 row(s) affected
✓	21	08:47:24	UPDATE `Turma_Estudante` SET `status_evasao` = 1 WHERE `estudante_matricula` = 2002	0 row(s) affected Rows matched
✓	22	08:47:30	SELECT * FROM `Log_Status_Evasao` LIMIT 0, 1000	0 row(s) returned
✓	23	08:47:46	SELECT * FROM `Log_Status_Evasao` LIMIT 0, 1000	0 row(s) returned
✓	24	08:48:55	UPDATE `Turma_Estudante` SET `status_evasao` = 1 WHERE `estudante_matricula` = 2002	0 row(s) affected Rows matched

Quinta Questão:



Obrigado pela Atenção!

Thank you for Watching!