

UNIVERSIDADE FEDERAL DE SÃO CARLOS
CAMPUS SOROCABA

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

BANCO DE DADOS
Prof. Sahudy Montenegro González

PROJETO PRÁTICO - FASE INTERMEDIÁRIA I/II

TEMA 12 - Banco de Questões de Disciplinas

Bianca Gomes Rodrigues
Pietro Zuntini Bonfim

Sorocaba-SP
2018

ÍNDICE

1	Descrição do Mini-Mundo	2
2	Projeto Conceitual	2
2.1	Modelo Entidade Relacionamento (MER)	2
2.2	Tipos-Entidade	3
2.3	Atributos dos Tipos-Entidade	3
2.4	Atributos Calculados	5
2.5	Tipos-Relacionamento	5
2.6	Requisitos de Dados	8
2.7	Tabelas de Metadados	10
3	Projeto Lógico	11
3.1	Mapeamento	11
3.2	Diagrama Entidade Relacionamento (DER)	13
4	Projeto Físico	14
4.1	Políticas de Restrição de Integridade	14
5	Especificação das Consultas em Álgebra Relacional e SQL	16
6	Trigger	22

1 DESCRIÇÃO DO MINI-MUNDO

Um dos problemas que muitos professores e alunos enfrentam na Universidade é não ter um mecanismo de aprendizado eficiente. Para tal, seria eficiente e interessante criar um sistema de gestão para armazenar questões de uma disciplina, de modo a auxiliar tanto os alunos quanto os professores, facilitando o controle do professor e o aprendizado e fixação dos alunos.

Através do sistema, é possível que o aluno tenha acesso à um banco de questões fornecido pelo professor de uma determinada disciplina, e possa estudar através deste para provas e outras avaliações respondendo às perguntas fornecidas e recebendo feedback de quantas questões respondeu e quantas acertou. Consequentemente, o professor também conseguiria ter controle do desempenho do aluno ao longo do semestre.

2 PROJETO CONCEITUAL

Começaremos descrevendo o Projeto Conceitual, ou seja, descrevendo como o problema do mini-mundo apresentado pode ser representado em uma abordagem de Banco de Dados. Nas subseções seguintes abordaremos as características, os atributos e os comportamentos das entidades envolvidas no sistema.

2.1 MODELO ENTIDADE RELACIONAMENTO (MER)

O Modelo Entidade Relacionamento (MER) descreve, em forma de diagrama, toda a especificação do problema, mostrando seus atores (Tipos-Entidade) e seus atributos, assim como os relacionamentos (Tipos-Entidade) entre os estes.

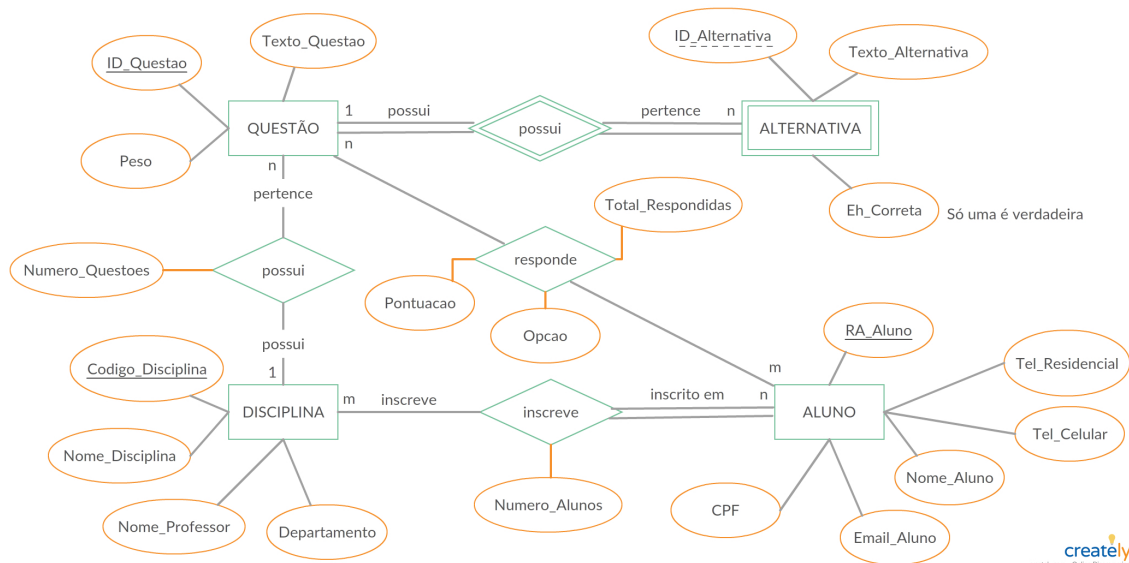


Figura 1: MER Completo

2.2 TIPOS-ENTIDADE

O sistema de Banco de Questões possui 4 tipos-entidade: *Disciplina*, *Aluno*, *Questão* e *Alternativa*. Notaremos no decorrer da próxima seção que todos os atributos são atômicos (simples) e monovalorados.

2.3 ATRIBUTOS DOS TIPOS-ENTIDADE

Nesta subseção serão descritos todos os tipos-entidade, junto com os atributos que os caracterizam.

DISCIPLINA

O tipo-entidade *Disciplina* é um objeto de existência conceitual e contém os atributos: *Codigo_Disciplina*, *Nome_Disciplina*, *Departamento* e *Nome_Professor*. O atributo *Codigo_Disciplina* é único e identifica o tipo-entidade. Também podemos identificar a disciplina pelo atributo *Nome_Disciplina*, entretanto este não é garantidamente único, podem existir disciplinas com o mesmo nome, lecionadas por diferentes professores.

O atributo *Departamento* é utilizado para armazenar o nome do departamento em que determinada *Disciplina* é lecionada. O nome de quem leciona determinada *Disciplina* é armazenado pelo atributo *Nome_Professor*. Abaixo será possível identificar as especificidades de cada atributo.

- *Codigo_Disciplina*: valor inteiro, auto incremental, chave primária, restrição obrigatório
- *Nome_Disciplina*: string de 255 caracteres, restrição obrigatório
- *Nome_Professor*: string de 255 caracteres, restrição obrigatório
- *Departamento*: string de 255 caracteres, restrição obrigatório

ALUNO

O tipo-entidade *Aluno* é um objeto de existência física e contém os atributos: *RA_Aluno*, *Nome_Aluno*, *CPF*, *Email_Aluno*, *Tel_Celular* e *Tel_Residencial*. O atributo *RA_Aluno* é único e identifica o tipo-entidade. É importante ressaltar este identificador é o *RA_Aluno* do aluno. Também é possível utilizar o atributo *Nome_Aluno* para encontrar determinado aluno, entretanto este não é um identificador único, visto que pode existir mais de um aluno com o mesmo nome.

O atributo *CPF* também é único, todavia neste caso não é utilizado para identificar um *Aluno*. Os atributos *Email_Aluno*, *Tel_Celular* e *Tel_Residencial*

servem para armazenar informações de contato sobre determinado **Aluno**. Abaixo será possível identificar as especificidades de cada atributo.

- **RA_Aluno**: valor inteiro, chave primária, restrição obrigatório
- **Nome_Aluno**: string de 255 caracteres, restrição obrigatório
- **CPF**: string de 14 caracteres, restrição obrigatório e único
- **Email_Aluno**: string de 50 caracteres, restrição obrigatório e único
- **Tel_Celular**: string de 15 caracteres, restrição obrigatório
- **Tel_Residencial**: string de 14 caracteres

QUESTÃO

O tipo-entidade **Questão** é um objeto de existência conceitual e contém os atributos: **ID_Questão**, **Texto_Questão** e **Peso**. O atributo **ID_Questão** é único e identifica o tipo-entidade. Já o atributo **Texto_Questão** contém o texto da questão que será apresentada ao aluno. Por último, o atributo **Peso** serve para mostrar o quanto vale cada questão e para o auxiliar no cálculo da pontuação do **Aluno**. Neste projeto, o **Peso** sempre será 1, ou seja, cada questão sempre valerá 1 ponto. Abaixo será possível identificar as especificidades de cada atributo.

- **ID_Questão**: valor inteiro, serial, chave primária, restrição obrigatório
- **Texto_Questão**: texto, restrição obrigatório
- **Peso**: valor inteiro, restrição obrigatório

ALTERNATIVA

O tipo-entidade **Alternativa** é um objeto de existência conceitual, além de ser um tipo-entidade fraca, logo, sua existência depende de sua participação no tipo-relacionamento com o tipo-entidade **Questão**. O tipo-entidade **Alternativa** contém os atributos: **ID_Alternativa**, **Texto_Alternativa** e **Eh_Correta**. O atributo **ID_Alternativa** é uma chave parcial e identifica o tipo-entidade em conjunto com o atributo **ID_Questão** do tipo-entidade **Questão**, visto que se trata de um tipo-entidade fraca. O atributo **Texto_Alternativa**, por sua vez, é responsável por armazenar o texto da alternativa, ou seja, uma possível resposta.

O atributo **Eh_Correta** é responsável por identificar se determinada alternativa está correta ou não. Uma alternativa precisa corresponder a uma determinada **Questão**, e apenas uma alternativa é correta dentre as alternativas de uma questão. Neste projeto, todas as questões possuem apenas quatro alternativas. Abaixo será possível identificar as especificidades de cada atributo.

- **ID_Alternativa:** valor inteiro, serial, chave parcial (ID_Questão e ID_Alternativa), restrição obrigatório
- **Texto_Alternativa:** texto, restrição obrigatório
- **Eh_Correta:** tupla ('Não', 'Sim'), restrição obrigatório

2.4 ATRIBUTOS CALCULADOS

O atributo **Total_Respondidas** é um atributo calculado e corresponde ao número total de questões respondidas pelo aluno. O atributo **Pontuação** também calculado e é obtido a cada questão que o aluno responde corretamente. É importante ressaltar que um aluno não perde pontos caso responda incorretamente determinada questão. Este atributo pode ajudar tanto o aluno, quanto o professor, através dele ambos podem acompanhar o nível de aprendizado. Ambos estes atributos são gerados pelo tipo-relacionamento **ALUNO responde QUESTÃO**.

O atributo **Número_Questões** é gerado através do tipo-relacionamento **DISCIPLINA possui QUESTÃO** e é responsável por armazenar o número de Questões existentes em uma determinada **Disciplina**. Por último, o atributo **Numero_Alunos** é gerado pelo tipo-relacionamento **DISCIPLINA inscreve ALUNO** e é responsável por armazenar o número de **Alunos** existentes em uma determinada **Disciplina**.

2.5 TIPOS-RELACIONAMENTO

ALUNO responde QUESTÃO

O tipo-entidade **Aluno** é responsável por responder questões do tipo-entidade **Questão**. Logo, o papel do tipo-relacionamento estabelecido entre eles é representado por **responde**. O tipo-relacionamento tem cardinalidade N:M, visto que um **Aluno** pode responder N **Questões**, e uma **Questão** é respondida por M **Alunos**.

O tipo-relacionamento **ALUNO responde QUESTÃO** possui o atributo **Opção**, além dos atributos calculados **Total_Respondidas** e **Pontuação**. O atributo **Opção** é responsável por armazenar qual foi a **Opção** respondida pelo **Aluno**.

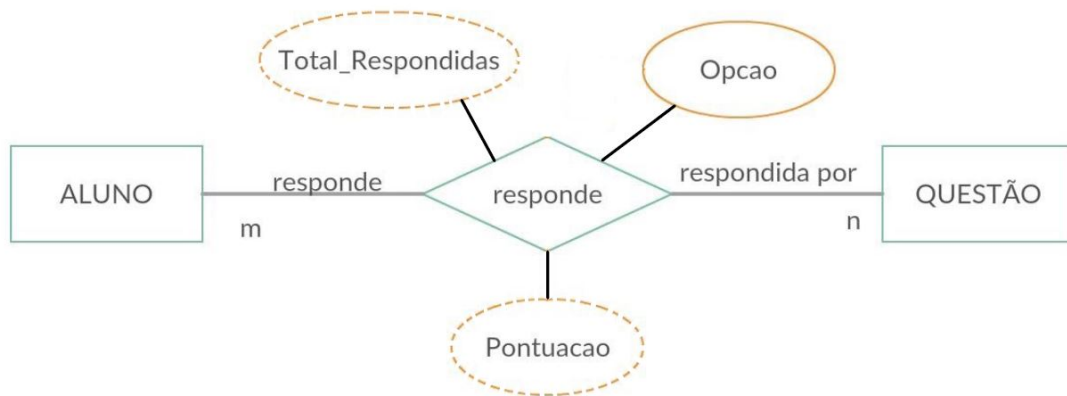


Figura 2: ALUNO responde QUESTÃO

DISCIPLINA inscreve ALUNO

O papel do tipo-relacionamento entre os tipo-entidades *Disciplina* e *Aluno* é representado por inscreve, visto que N *Alunos* podem se inscrever em uma *Disciplina*, e um *Aluno* pode estar inscrito em M *Disciplinas*. Deste modo, podemos concluir que a cardinalidade deste tipo-relacionamento é 1:N. Além disso, este tipo-relacionamento possui o atributo calculado *Numero_Alunos*.

A existência do tipo-entidade *Disciplina* depende de sua participação em um tipo-relacionamento com o do tipo-entidade *Aluno*. Logo, se trata de uma restrição de participação total, uma *Disciplina* existirá apenas caso hajam *Alunos* inscritos.

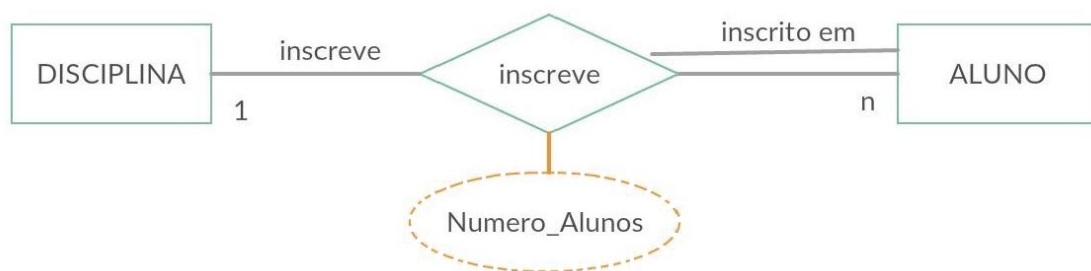


Figura 3: DISCIPLINA inscreve ALUNO

DISCIPLINA possui QUESTÃO

O papel do tipo-relacionamento entre os tipo-entidades *Disciplina* e *Questão* é representado por *possui*, visto que uma *Disciplina*, pode possuir N *Questões*, e uma *QUESTÃO* pertence apenas a uma *DISCIPLINA*. Deste modo, podemos concluir que a cardinalidade deste tipo-relacionamento é 1:N. Além disso, este é o tipo-relacionamento que possui o atributo calculado *Numero_Questoes*.

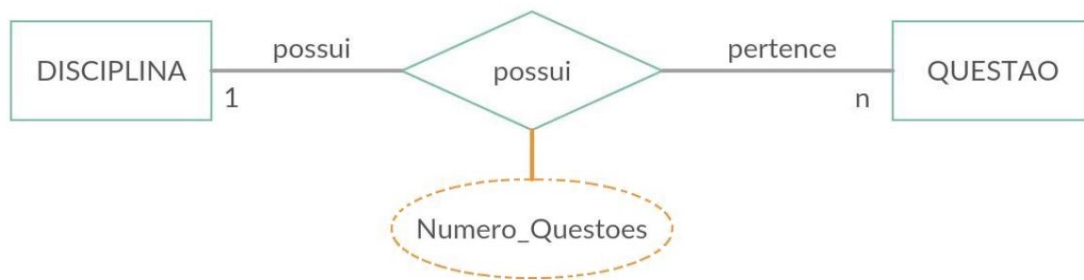


Figura 4: DISCIPLINA possui QUESTÃO

QUESTÃO possui ALTERNATIVA

O tipo-relacionamento entre os tipo-entidades *Questão* e *Alternativa* é representado por *possui*, visto que uma *Questão* possui N *Alternativas*. Do mesmo modo, N *Alternativas* pertencem à uma *Questão*. Logo, concluímos que a cardinalidade do tipo-relacionamento é 1:N. Ressaltando, neste projeto N sempre será igual a quatro.

A existência do tipo-entidade *Questão* depende de sua participação em um tipo-relacionamento com o do tipo-entidade *Alternativa*. Logo, se trata de uma restrição de participação total, só existe se uma *Questão* possuir *Alternativa(s)*.

O tipo-entidade *Alternativa* é fraco, logo sua existência depende de sua participação no tipo-relacionamento com o tipo-entidade *Questão*, que a identifica. Uma *Alternativa* existirá apenas caso exista uma *Questão* correspondente.



Figura 5: QUESTÃO possui ALTERNATIVA

2.6 REQUISITOS DE DADOS

Os Requisitos de Dados representam as consultas, inserções, modificações, remoções, agregações que o projeto deve ser capaz de executar. A seguir será possível visualizar alguns exemplos:

INSERÇÕES

1. O sistema deve permitir o cadastro de um **Aluno** em uma **Disciplina** a partir dos atributos do tipo-entidade **Aluno**
2. O sistema deve permitir o cadastro de uma **Disciplina** a partir dos atributos do tipo-entidade **Disciplina**
3. O sistema deve permitir o cadastro de uma **Questão** de uma determinada **Disciplina** a partir dos atributos do tipo-entidade **Questão**
4. O sistema deve permitir a adição de **Alternativas** de uma determinada **Questão** a partir dos atributos do tipo-entidade **Alternativa**

BUSCAS

1. O sistema deve permitir a busca de um **Aluno** pertencente à uma **Disciplina**
 - Atributo(s) de visualização do resultado: **RA_Aluno**, **Nome_Aluno**
 - Atributo(s) de busca (ou de condições/filtros): **RA_Aluno** ou **Nome_Aluno**, **Codigo_Disciplina**
2. O sistema deve permitir busca de **Questões** a partir de uma palavra chave
 - Atributo(s) de visualização do resultado: **ID_Questão**, **Texto_Questão**, **Codigo_Disciplina**
 - Atributo(s) de busca (ou de condições/filtros): palavra chave ('%palavra-chave%')

3. O sistema deve permitir busca de Questões de uma determinada *Disciplina*
 - Atributo(s) de visualização do resultado: *ID_Questão*, *Texto_Questão*
 - Atributo(s) de busca (ou de condições/filtros): *Codigo_Disciplina*
4. O sistema deve permitir busca de uma *Disciplina* à partir de seu código
 - Atributo(s) de visualização do resultado: atributos da *Disciplina* encontrada
 - Atributo(s) de busca (ou de condições/filtros): *Codigo_Disciplina*
5. O sistema deve permitir o cálculo do número de *Alunos* inscritos por *Disciplina*
 - Atributo(s) de visualização do resultado: número de alunos
 - Atributo(s) de cálculo: *Codigo_Disciplina*
6. O sistema deve permitir o cálculo do número de Questões respondidas por um *Aluno*
 - Atributo(s) de visualização do resultado: número de questões respondidas
 - Atributo(s) de cálculo: obtido através do tipo-relacionamento *ALUNO responde QUESTÃO*
7. O sistema deve permitir o cálculo da pontuação do *Aluno*
 - Atributo(s) de visualização do resultado: pontuação do *Aluno*
 - Atributo(s) de cálculo: peso da questão e número de acertos
8. O sistema deve permitir o cálculo do número de Questões de uma determinada *Disciplina*
9. O sistema deve permitir a visualização de todas as alternativas de uma determinada questão
10. O sistema deve gerar relatórios da média da pontuação dos *Alunos* por *Disciplina*
11. O sistema deve gerar relatórios da média do número de Questões respondidas por *Disciplina*
12. O sistema deve gerar relatórios do número de Questões respondidas por *Aluno*

MODIFICAÇÕES

1. O sistema deve permitir a alteração de dados da *Disciplina*
2. O sistema deve permitir a alteração de dados do *Aluno*

REMOÇÕES

1. O sistema deve permitir a remoção de Questões
2. O sistema deve permitir a remoção de Alunos

2.7 TABELAS DE METADADOS

A tabela de metadados contém basicamente um resumo de cada tipo-entidade, com seu nome e atributos, junto com o tipo e restrição de cada atributo.

Tabela 1: Tipo-Entidade Aluno

Tipo-Entidade	Atributo	Tipo	Restrição
Aluno	RA_Aluno	Identificador	Obrigatório
	Nome_Aluno	Monovalorado	Obrigatório
	CPF	Monovalorado	Obrigatório e Único
	Email_Aluno	Monovalorado	Obrigatório e Único
	Tel_Celular	Monovalorado	Obrigatório
	Tel_Residencial	Monovalorado	Opcional

Tabela 2: Tipo-Entidade Disciplina

Tipo-Entidade	Atributo	Tipo	Restrição
Disciplina	Codigo_Disciplina	Identificador	Obrigatório
	Nome_Disciplina	Monovalorado	Obrigatório
	Nome_Professor	Monovalorado	Obrigatório
	Departamento	Monovalorado	Obrigatório

Tabela 3: Tipo-Entidade Questão

Tipo-Entidade	Atributo	Tipo	Restrição
Questão	ID_Questao	Identificador	Obrigatório
	Texto_Questao	Monovalorado	Obrigatório
	Peso	Monovalorado	Obrigatório, Peso = 1

Tabela 4: Alternativa

Tipo-Entidade	Atributo	Tipo	Restrição
Alternativa	ID_Alternativa	Chave Parcial	Obrigatório
	Texto_Alternativa	Monovalorado	Obrigatório
	Eh_Correta	Monovalorado	Obrigatório, 'Não' ou 'Sim'

3 PROJETO LÓGICO

O projeto lógico consiste no mapeamento do esquema desenvolvido na seção **Projeto Conceitual**, além da aplicação das regras de normalização que são 1FN, 2FN e 3FN.

3.1 MAPEAMENTO

Inicialmente os tipo-entidade forte **Questão**, **Disciplina** e **Aluno** foram mapeados. Para isso utilizamos n colunas, uma para cada atributo. Por conseguinte, foram obtidas as seguintes relações:

```
Questao (id_questao, texto_questao, peso)
```

```
Disciplina (codigo_disciplina, nome_disciplina,  
departamento, nome_professor)
```

```
Aluno (ra_aluno, cpf, nome_aluno, email_aluno,  
tel_residencial, tel_celular)
```

Em seguida, o tipo-entidade fraca **Alternativa** foi mapeado, criando uma tabela própria com chave primária igual ao conjunto dos dois tipos-entidade envolvidos: **Questão** e **Alternativa**. Por conseguinte, foi obtida a seguinte relação:

```
Alternativa (id_questao, id_alternativa, texto_alternativa,  
eh_correta)  
id_questao referencia Questao
```

Em seguida, foram mapeados os tipos-relacionamento 1:n.

Para o relacionamento **DISCIPLINA** possui **QUESTÃO**, foi utilizada a técnica de adição de coluna, criando um atributo no lado n, atualizando o mapeamento da **Questão**:

```
Questao (id_questao, texto_questao, peso,  
codigo_disciplina)  
codigo_disciplina referencia Disciplina
```

Além disso, foi inserido o atributo calculado **Numero_Questoes**, atualizando o mapeamento da **Disciplina**:

```
Disciplina (codigo_disciplina, nome_disciplina,  
departamento, nome_professor, numero_questoes)
```

Para tipo-relacionamento **QUESTÃO** possui **ALTERNATIVA**, mapeamos novamente através da técnica de adição de coluna, criando um atributo do lado n. Entretanto, se trata de uma entidade fraca, logo, a relação permaneceu igual:

```
Alternativa (id_questao, id_alternativa, texto_alternativa,  
eh_correta)  
id_questao referencia Questao
```

Por fim, foram mapeados os tipos-relacionamento n:m.

Para o tipo-relacionamento ALUNO responde QUESTÃO, foi utilizada a técnica de criação de uma tabela própria, obtendo a seguinte relação:

```
Responde (ra_aluno, id_questao, opcao)
ra_aluno referencia Aluno
id_questao referencia Questao
```

Além disso, foram inseridos os atributos calculados Total_Respondidas e Pontuacao, atualizando o mapeamento do Aluno:

```
Aluno (ra_aluno, cpf, nome_aluno, email_aluno,
tel_residencial, tel_celular, total_respondidas, pontuacao)
```

Para o tipo-relacionamento DISCIPLINA inscreve ALUNO, foi utilizada a técnica de criação de tabela própria, obtendo a seguinte relação:

```
Aluno_Disciplina (ra_aluno, codigo_disciplina)
ra_aluno referencia Aluno
codigo_disciplina referencia Disciplina
```

Além disso, foi inserido o atributo calculado Numero_Alunos, atualizando o mapeamento da Disciplina:

```
Disciplina (codigo_disciplina, nome_disciplina,
departamento, nome_professor, numero_questoes, numero_alunos)
```

Para melhor visualização, mostraremos a seguir todas as relações criadas:

```
Disciplina (codigo_disciplina, nome_disciplina,
departamento, nome_professor, numero_questoes, numero_alunos)
```

```
Aluno (ra_aluno, cpf, nome_aluno, email_aluno,
tel_residencial, tel_celular, total_respondidas, pontuacao)
```

```
Aluno_Disciplina (ra_aluno, codigo_disciplina)
ra_aluno referencia Aluno
codigo_disciplina referencia Disciplina
```

```
Questao (id_questao, texto_questao, peso,
codigo_disciplina)
codigo_disciplina referencia Disciplina
```

```
Alternativa (id_questao, id_alternativa, texto_alternativa,
eh_correta)
id_questao referencia Questao
```

```
Responde (ra_aluno, id_questao, opcao)
ra_aluno referencia Aluno
id_questao referencia Questao
```

É importante ressaltar que mapeamento se encontra na 1FN, visto que todos os atributos são atômicos e monovalorados, ou seja, não existem atributos multivalorados. Também podemos dizer que o mapeamento se encontra na 2FN, visto que se encontra na 1FN e não há dependência funcional parcial.

Por fim, também podemos dizer que os mapeamentos se encontram na 3FN, visto que estão na 1FN, 2FN e nenhum atributo não chave depende de atributos não chave.

3.2 DIAGRAMA ENTIDADE RELACIONAMENTO (DER)

Obtidas todas as relações e verificadas as normalizações, foi possível obter o diagrama completo com todas as entidades, além de setas indicando as referências às outras entidades (chaves estrangeiras).

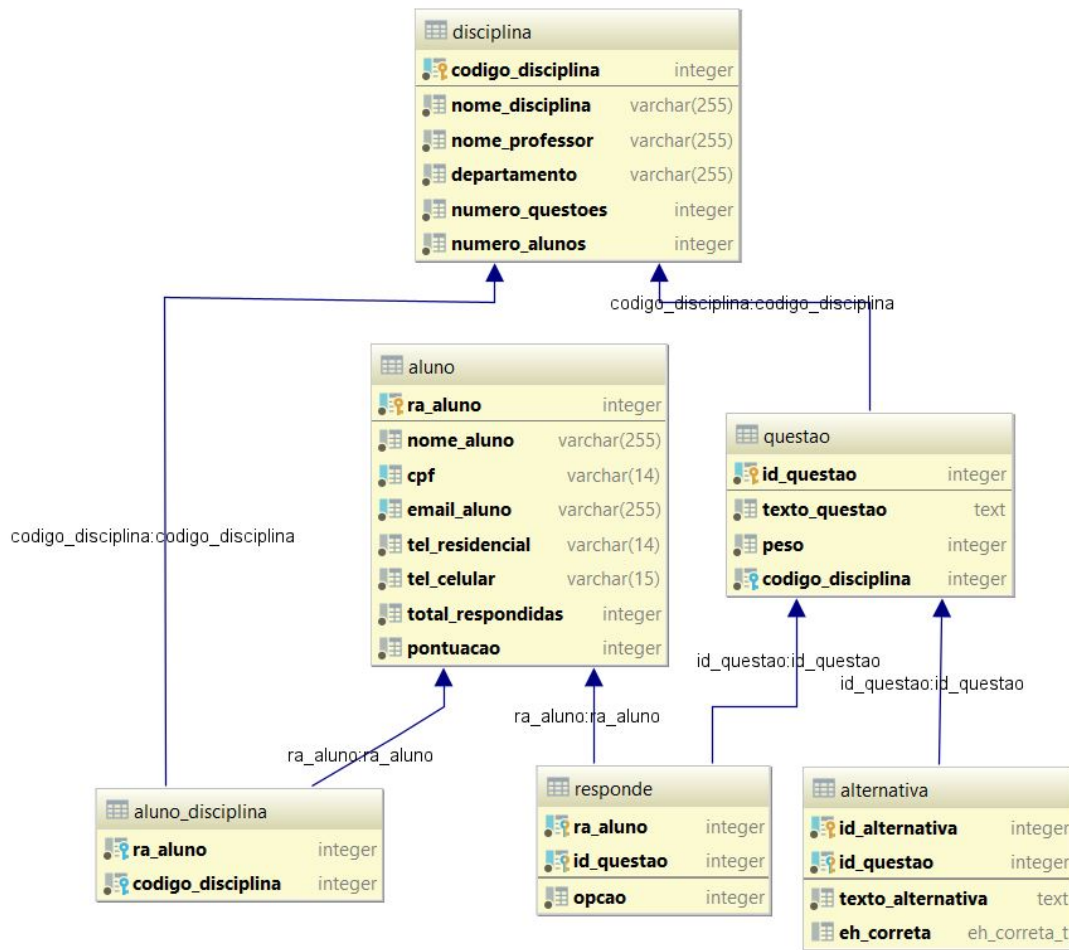


Figura 6: DER Completo

4 PROJETO FÍSICO

O projeto físico corresponde à implementação em si do Banco de Dados Relacional. Os comandos foram criados à partir da linguagem SQL, através do Sistema de Gerenciamento de Banco de Dados (SGBD) PostgreSQL. Os scripts de criação do banco de dados se encontram anexados no arquivo *esquema.sql*, e os de inserção se encontram anexados no arquivo *dados.sql*;

4.1 POLÍTICAS DE RESTRIÇÃO DE INTEGRIDADE

Para a tabela *Disciplina*, temos as seguintes restrições de integridade:

- *Codigo_Disciplina*: NOT NULL e PRIMARY KEY
- *Nome_Disciplina*: NOT NULL

- Nome_Professor: NOT NULL
- Departamento: NOT NULL
- Numero_Questoes: NOT NULL e DEFAULT 0
- Numero_Alunos: NOT NULL, DEFAULT 0 e CHECK (Numero_Alunos <= 20)

Para a tabela Aluno, temos as seguintes restrições de integridade:

- RA_Aluno: NOT NULL e PRIMARY KEY
- Nome_Aluno: NOT NULL
- CPF: NOT NULL e UNIQUE
- Email_Aluno: NOT NULL e UNIQUE
- Tel_Celular: NOT NULL
- Tel_Residencial:
- Total_Respondidas: NOT NULL e DEFAULT 0
- Tel_Pontuacao: NOT NULL e DEFAULT 0

Para a tabela Aluno_Disciplina, temos as seguintes restrições de integridade:

- RA_Aluno: NOT NULL, PRIMARY KEY, FOREIGN KEY, ON DELETE CASCADE e ON UPDATE CASCADE
- Codigo_Disciplina: NOT NULL, PRIMARY KEY, FOREIGN KEY, ON DELETE CASCADE e ON UPDATE CASCADE

Para a tabela Questao, temos as seguintes restrições de integridade:

- ID_Questão: NOT NULL e PRIMARY KEY
- Texto_Questão: NOT NULL
- Peso: NOT NULL, DEFAULT 1 e CHECK (Peso = 1)
- Codigo_Disciplina: NOT NULL, FOREIGN KEY, ON DELETE RESTRICT e ON UPDATE CASCADE

Para a tabela Alternativa, temos as seguintes restrições de integridade:

- ID_Alternativa: NOT NULL e PRIMARY KEY

- ID_Questao: NOT NULL, PRIMARY KEY, FOREIGN KEY, ON DELETE CASCADE e ON UPDATE CASCADE
- Texto_Alternativa: NOT NULL
- Eh_Correta: 'Sim' ou 'Nao'

Para a tabela **Responde**, temos as seguintes restrições de integridade:

- ID_Alternativa: NOT NULL e PRIMARY KEY
 - ID_Questao: NOT NULL, PRIMARY KEY, FOREIGN KEY, ON DELETE CASCADE e ON UPDATE CASCADE
 - Texto_Alternativa: NOT NULL
 - Eh_Correta: 'Sim' ou 'Nao'
-
- RA_Aluno: NOT NULL, PRIMARY KEY, FOREIGN KEY, ON DELETE CASCADE e ON UPDATE CASCADE
 - ID_Questao: NOT NULL, PRIMARY KEY, FOREIGN KEY, ON DELETE CASCADE e ON UPDATE CASCADE
 - Opcao: NOT NULL

5 ESPECIFICAÇÃO DAS CONSULTAS EM ÁLGEBRA RELACIONAL E SQL

Nesta seção serão apresentadas as consultas propostas previamente na seção **Requisitos de Dados**. As consultas a seguir estão tanto na Álgebra Relacional (AR), como na linguagem do banco de dados SQL.

Consulta 1 - Buscar um determinado Aluno pertencente a uma determinada Disciplina

AR

$\text{join} \leftarrow \text{aluno} \bowtie \text{aluno_disciplina}$

$\text{linhas} \leftarrow \sigma_{(\text{RA_Aluno}=\langle \text{RA_Aluno} \rangle \wedge \text{Codigo_Disciplina}=\langle \text{Codigo_Disciplina} \rangle)}(\text{join})$

$\pi_{(\text{RA_Aluno}, \text{Nome_Aluno})}(\text{linhas})$

SQL

```
SELECT RA_Aluno, Nome_Aluno
FROM aluno
JOIN aluno_disciplina
ON aluno.RA_Aluno = aluno_disciplina.RA_Aluno
WHERE aluno.RA_Aluno = <RA_Aluno>
AND Codigo_Disciplina = <Codigo_Disciplina>;
```

Consulta 2 - Listar uma determinada Questão e suas respectivas Alternativas a partir de uma palavra chave

AR

```
join  $\leftarrow$  questao  $\bowtie$  alternativa

linhas  $\leftarrow \sigma_{\text{Texto\_Questao} = \%<\text{palavrachave}>\%}(join)$ 

 $\pi_{(ID\_Questao, \text{Texto\_Questao}, \text{Codigo\_Disciplina})}(linhas)$ 
```

SQL

```
SELECT *
FROM questao
JOIN alternativa
ON questao.ID_Questao = alternativa.ID_Questao
WHERE Texto_Questao ILIKE '%<palavrachave>%';
```

Consulta 3 - Listar todas as Questões de uma determinada Disciplina

AR

```
linhas  $\leftarrow \sigma_{\text{Codigo\_Disciplina} = <\text{Codigo\_Disciplina}>}(questao)$ 

 $\pi_{(ID\_Questao, \text{Texto\_Questao}, \text{Codigo\_Disciplina})}(linhas)$ 
```

SQL

```
SELECT *
FROM questao
WHERE Codigo_Disciplina = <Codigo_Disciplina>;
```

Consulta 4 - Listar as Disciplinas a partir do código

AR

$\text{linhas} \leftarrow \sigma_{(\text{Codigo_Disciplina}=\langle\text{Codigo_Disciplina}\rangle)}(\text{disciplina})$

$\pi_{(\text{Codigo_Disciplina}, \text{Nome_Disciplina}, \text{Nome_Professor}, \text{Departamento}, \text{Numero_Questoes}, \text{Numero_Alunos})}(\text{linhas})$

SQL

```
SELECT *  
FROM disciplina  
WHERE Codigo_Disciplina = <Codigo_Disciplina>;
```

Consulta 5 - Calcula o número de Alunos inscritos por Disciplina

AR

$\text{temp} \leftarrow_{\text{Codigo_Disciplina}} F_{(\text{COUNT}(\text{RA_Aluno}))}(\text{aluno_disciplina})$

$\rho_{(\text{Codigo_Disciplina}, \text{Numero_Alunos})}(\text{temp})$

SQL

```
SELECT aluno_disciplina.Codigo_Disciplina AS Codigo_Disciplina,  
       count(aluno_disciplina.RA_Aluno) AS Numero_Alunos  
FROM aluno_disciplina  
GROUP BY aluno_disciplina.Codigo_Disciplina;
```

Consulta 6 - Calcula o número de questões respondidas por um determinado Aluno

AR

$\text{temp}_1 \leftarrow \sigma_{(\text{RA_Aluno}=\langle\text{RA_Aluno}\rangle)}(\text{responde})$

$\text{temp}_2 \leftarrow F_{(\text{COUNT}(\text{RA_Aluno}))}(\text{temp}_1)$

$\rho_{(\text{Numero_Questoes})}(\text{temp}_2)$

SQL

```
SELECT count(RA_Aluno) AS Numero_Questoes  
FROM responde  
WHERE RA_Aluno = <RA_Aluno>;
```

Consulta 7 - Calcula a pontuação do Aluno

AR

```
join  $\leftarrow$  responde  $\bowtie$  alternativa  
linha  $\leftarrow \sigma_{(RA\_Aluno=<RA\_Aluno>\wedge Eh\_Correta="Sim")}(join)$   
temp1  $\leftarrow F_{(COUNT(RA\_Aluno))}(linha)$   
 $\rho_{(Pontuacao)}(temp_1)$ 
```

SQL

```
SELECT count(RA_Aluno) AS Pontuacao  
FROM responde  
JOIN alternativa  
ON responde.Opcao = alternativa.ID_Alternativa  
WHERE RA_Aluno = <RA_Aluno> AND alternativa.Eh_Correta = 'Sim';
```

Consulta 8 - Calcula o número de Alunos

AR

```
temp1  $\leftarrow \sigma_{(Codigo\_Disciplina=<Codigo\_Disciplina>)}(questao)$   
temp2  $\leftarrow F_{(COUNT(ID\_Questao))}(temp_1)$   
 $\rho_{(Numero\_Questoes)}(temp_2)$ 
```

SQL

```
SELECT count(ID_Questao) AS Numero_Questoes  
FROM questao  
WHERE Codigo_Disciplina = <Codigo_Disciplina>;
```

Consulta 9 - Lista todas as Alternativas de uma determinada Questão

AR

```
join  $\leftarrow$  questao  $\bowtie$  alternativa  
linhas  $\leftarrow \sigma_{(ID\_Questao=<ID\_Questao>)}(join)$   
 $\pi_{(Texto\_Questao, Texto\_Alternativa, Eh\_Coreta)}(linhas)$ 
```

SQL

```
SELECT Texto_Questao, Texto_Alternativa, Eh_Correta
FROM questao
JOIN alternativa
ON questao.ID_Questao = alternativa.ID_Questao
WHERE questao.ID_Questao = <ID_Questao>;
```

Consulta 10 - Lista todos os Alunos de uma determinada Disciplina

AR

```
join  $\leftarrow$  aluno  $\bowtie$  aluno_disciplina

linhas  $\leftarrow \sigma_{(\text{Codigo\_Disciplina}=\text{<Codigo\_Disciplina>}}(\text{join})$ 

 $\pi_{(\text{Codigo\_Disciplina}, \text{RA\_Aluno}, \text{Nome\_Aluno})}(\text{linhas})$ 
```

SQL

```
SELECT aluno_disciplina.Codigo_Disciplina,
       aluno.RA_Aluno, aluno.Nome_Aluno
FROM aluno_disciplina
JOIN aluno
ON aluno_disciplina.RA_Aluno = aluno.RA_Aluno
WHERE Codigo_Disciplina = <Codigo_Disciplina>;
```

Consulta 11 - Calcula a média da pontuação dos Alunos por Disciplina

AR

```
join  $\leftarrow$  aluno  $\bowtie$  aluno_disciplina

temp  $\leftarrow_{\text{Codigo\_Disciplina}} F_{(\text{AVG}(\text{Pontuacao}))}(\text{join})$ 

 $\rho_{(\text{Codigo\_Disciplina}, \text{Media\_Pontuacao})}(\text{temp})$ 
```

SQL

```
SELECT aluno_disciplina.codigo_disciplina,
       avg(pontuacao) AS Media_Pontuacao
FROM aluno_disciplina
JOIN aluno
ON aluno.RA_Aluno = aluno_disciplina.RA_Aluno
GROUP BY aluno_disciplina.Codigo_Disciplina
ORDER BY Media_Pontuacao
```

Consulta 12 - Calcula a média de Questões respondidas por Disciplina

AR

```
join  $\leftarrow$  aluno  $\bowtie$  aluno_disciplina  
temp  $\leftarrow$   $\text{Codigo\_Disciplina}$   $F_{(AVG(Total\_Respondidas))}(join)$   
 $\rho_{(Codigo\_Disciplina, Media\_Respondidas)}(temp)$ 
```

SQL

```
SELECT codigo_disciplina,  
       avg(total_respondidas) AS Media_Respondidas  
FROM aluno_disciplina  
JOIN aluno  
ON aluno_disciplina.ra_aluno = aluno.ra_aluno  
GROUP BY aluno_disciplina.codigo_disciplina  
ORDER BY Media_Respondidas
```

Consulta 13 - Listar o número de questões respondidas por Aluno

AR

```
temp  $\leftarrow$   $RA\_Aluno$   $F_{(COUNT(RA\_Aluno))}(responde)$   
 $\rho_{(RA\_Aluno, Numero\_Questoes)}(temp)$ 
```

SQL

```
SELECT RA_Aluno, count(RA_Aluno) AS Numero_Questoes  
FROM responde  
GROUP BY ra_aluno;
```

6 TRIGGER

A trigger criada atualiza a pontuação do Aluno a cada inserção na tabela Responde. A atualização é realizada após a conferência do atributo Eh_Correta, caso esteja de acordo com a resposta esperada, 'Sim', então há o aumento, em um ponto, da pontuação geral do Aluno. Apenas um ponto é aumentado, visto que neste projeto definimos que o peso é sempre igual a um.

```
01 | -- Criacao do PROCEDIMENTO
02 | CREATE FUNCTION atualizar_pontuacao() RETURNS TRIGGER AS
    $atualizar_pontuacao$
03 |     DECLARE
04 |         resultado Eh_Correta_t;
05 |     BEGIN
06 |
07 |         -- Coloca a verificacao em 'resultado'
08 |         SELECT Eh_Correta INTO resultado FROM alternativa
09 |         WHERE ID_Alternativa = NEW.Opcao;
10 |
11 |         -- Compara se e correta
12 |         IF (resultado = 'Sim') THEN
13 |
14 |             -- Atualiza a pontuacao do Aluno
15 |             UPDATE aluno SET Pontuacao = (Pontuacao + 1)
16 |             WHERE aluno.RA_Aluno = NEW.RA_Aluno;
17 |
18 |             RETURN NEW;
19 |         END IF;
20 |
21 |         -- Caso nao esteja correta, nao modifica a pontuacao
22 |         RETURN NULL;
23 |     END;
24 | $atualizar_pontuacao$ LANGUAGE plpgsql;
25 |
26 | -- Criacao da TRIGGER
27 | CREATE TRIGGER atualizar_pontuacao
28 | AFTER INSERT
29 | ON responde
30 | FOR EACH ROW EXECUTE PROCEDURE atualizar_pontuacao();
```