Levantamento de requisitos e modelagem de um banco de dados para um sistema de gerenciamento de bibliotecas

Roberval Requião Junior e Gabriel Andrade de Araujo

1. Introdução

Para uma boa modelagem de um banco de dados de qualquer domínio, é necessário um estudo visando o entendimento do seu escopo de funcionamento, de forma que as decisões tomadas atendam todas as suas necessidades, da melhor forma possível. Com isso em mente, foram realizados neste trabalho levantamentos e pesquisas, permitindo uma familiarização com o domínio escolhido e seus requisitos.

O banco de dados deste trabalho foi desenvolvido com o objetivo de permitir o gerenciamento de bibliotecas. Um projeto deste tipo demanda um sistema que permita incluir, principalmente, informações sobre livros, usuários e bibliotecários, assim como informações sobre empréstimos, devoluções e multas. Outras informações relevantes para esta área são em relação a categorias, seções, estantes, endereços, editoras, exemplares e autores.

2. Levantamento de requisitos

Para o levantamento de requisitos deste sistema, foi pensado primeiro no cadastro de usuários e bibliotecários. Após pesquisas realizadas acerca de dados cadastrais comumente utilizados em diversos ambitos, foi criada a seguinte tabela, reunindo os dados mais importantes identificados:

Pessoa	
Campo	**Descrição**
Nome	Nome completo
Telefone	Telefone para contato
Endereços	Endereços residenciais ou comerciais
Nascimento	Data de nascimento
RG	Número do Registro Geral
CPF	Número do Cadastro de Pessoa Física
E-mail	Endereço de e-mail para contato

Tabela 1 – Dados de pessoas

No campo de endereços, é necessário armazenar seguintes dados:

Endereço	
Campo	**Descrição**
Rotulo	Endereço residencial, endereço comercial etc.
Cep	Código de endereçamento postal
Logradouro	Rua, avenida etc.
Número	Número da residência, edifício etc.
Complemento	Código do bloco, apartamento, casa etc.
Bairro	Nome do bairro
Cidade	Nome da cidade
Estado	Nome do estado

Tabela 2 – Dados de endereço

Após isso, foram levantados os dados referentes ao cadastro de livros, com consultas realizadas em bibliotecas físicas e virtuais, onde identificou-se que, comumente, um livro se cadastra de duas formas simultâneas, como livro e como exemplar. O livro refere-se aos seus dados “virtuais”, enquanto o exemplar refere-se ao seu objeto físico, que é o que de fato será disponibilizado, movimentado, emprestado, devolvido etc., representando uma cópia física do livro. Foram então levantados os seguintes dados:

Livro	
Campo	**Descrição**
ISBN	*International Standard Book Number*
Data de publicação	Data de publicação
Título	Título do livro
Idioma	Idioma do livro
Autor	Autor do livro
Editora	Editora que publicou o livro
Categoria	Categoria do livro

Tabela 3 – Dados de livros

Exemplar	
Campo	**Descrição**
Estante	Estante onde é armazenado
Data de aquisição	Data de aquisição do exemplar
Descrição	Descrição ou observação para registro
Livro	Dado identificador do livro que o exemplar representa

Tabela 4 – Dados de exemplares

Para os dados de autores, estantes, seções, editora e categoria, como são objetos que não demandam muitos dados, objetivando apenas registro e organização, foi filtrado apenas um campo para cada, sendo um campo chamado “nome” para os autores, seções, editora e categoria e, para a estante, um campo chamado local, que descreve o local onde ela se encontra.

Por fim, foram reunidos os dados referentes a realização de empréstimos, que geram também dados de devoluções e multas, caso ocorram. Estes dados foram levantados por meio da observação do funcionamento de bibliotecas físicas e virtuais, além de pesquisas em websites, obtendo-se as informações abaixo:

Empréstimo	
Campo	**Descrição**
Data do empréstimo	Data de efetuação do empréstimo
Hora do empréstimo	Hora de efetuação do empréstimo
Data final	Data final para a devolução
Nome do usuário	Código do usuário atendido
Nome do bibliotecario	Código do bibliotecario que atendeu
Exemplar	Exemplar emprestado

Tabela 4 – Dados de empréstimos

Devolução	
Campo	**Descrição**
Data de devolução	Data de efetuação da devolução
Hora de devolução	Hora de efetuação da devolução

Tabela 5 – Dados de devoluções

Multa	
Campo	**Descrição**
Dias de atraso	Número de dias de atraso
Valor	Valor total da multa
Data de pagamento	Data de efetuação do pagamento
Hora de pagamento	Hora de efetuação do pagamento

Tabela 5 – Dados de multas

3. Elaboração dos diagramas

Após o entendimento do problema e o levantamento dos requisitos, foi possível elaborar os diagramas, que auxiliam no planejamento de um sistema, permitindo definir como ele deverá ficar, fazendo com que o projeto siga de forma organizada, podendo-se também identificar possíveis problemas de planejamento e corrigí-los antes da etapa de desenvolvimento.

O primeiro diagrama elaborado foi o Diagrama Entidade-Relacionamento, que é um tipo de fluxograma que ilustracomo as “entidades” do banco de dados, tais como pessoas, objetos ou conceitos,se relacionam entre si.Para isso, utiliza-se um conjunto definido de símbolos, tais como retângulos, diamantes, ovais e linhas de conexão para representar a interconectividade de entidades, relacionamentos e seus atributos. O diagrama encontra-se disponível no Anexo I deste documento.

O segundo diagrama desenvolvido foi o Diagrama de Classes, que tem como objetivo principal a especificação dos componentes de um sistema e suas interligações, em relação à sua estrutura, sendo bastante utilizado nas etapas de design ou esboço de ideias de um software. O diagrama encontra-se disponível no Anexo II deste documento.

O terceiro diagrama desenvolvido foi o Modelo Relacional, que é um modelo de dados representativo, servindo como o modelo subjacente de um sistema de banco de dados, baseando-se no princípio do armazenamento de dados em tabelas. O diagrama encontra-se disponível no Anexo III deste documendo, onde é possível verificar que ele está na terceira forma normal, pois:

- Todos os atributos das tabelas são atômicos, não contendo grupos repetidos e nem atributos com mais de um valor, cumprindo a primeira forma normal;
- Todos os atributos não chaves das tabelas dependem totalmente da chave primária, sem ocorrência de dependências parciais, cumprindo a segunda forma normal;
- Todos os atributos não chave das tabelas são mutualmente independentes, dependendo unicamente e exclusivamente da chave primária, cumprindo a terceira forma normal.

4. Elaboração do Dicionário de Dados

Após a modelagem e construção dos diagramas descritos, foi elaborado o dicionário de dados, que auxilia no planejamento e define a forma como os dados deverão ser tratados pelo sistema. Para isso, foram construídas as tabelas abaixo:

Entidade: autor					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_autor	Determinante	Numérico		Serial	
nome	Simples	Texto	255		Nome completo

Entidade: bibliotecario					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_bibliotecario	Determinante	Numérico		Serial	
nome	Simples	Texto	255		Nome completo
telefone	Simples	Texto			Telefone com formato livre
nascimento	Simples	Data			Data de nascimento
rg	Simples	Texto	12		Rg com formato xx.xxx.xxx-x
cpf	Simples	Texto	14		CPF com formato xxx.xxx.xxx-xx
email	Simples	Texto	100		Endereço de e-mail

Entidade: usuario					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_usuario	Determinante	Numérico		Serial	
nome	Simples	Texto	255		Nome completo
telefone	Simples	Texto			Telefone com formato livre
nascimento	Simples	Data			Data de nascimento
rg	Simples	Texto	12		Rg com formato xx.xxx.xxx-x
cpf	Simples	Texto	14		CPF com formato xxx.xxx.xxx-xx
email	Simples	Texto	100		Endereço de e-mail

Entidade: endereco_usuario					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_endereco_usuario	Determinante	Numérico			
codigo_usuario	Determinante	Numérico			Código do usuário
rotulo	Simples	Texto	50		Endereço residencial, endereço comercial etc.
cep	Simples	Texto	10		CEP com formato xx.xxx-xxx
logradouro	Simples	Texto	150		Rua, avenida etc.
numero	Simples	Numérico			Número da residencia, edifício etc.
complemento	Simples	Texto	50		Código do bloco, apartamento, casa etc.
bairro	Simples	Texto	50		Nome do bairro
cidade	Simples	Texto	50		Nome da cidade
estado	Simples	Texto	2		Sigla do estado com 2 dígitos

Entidade: endereco_bibliotecario					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_endereco_bibliotecario	Determinante	Numérico			
codigo_usuario	Determinante	Numérico			
rotulo	Simples	Texto	50		Endereço residencial, endereço comercial etc.
cep	Simples	Texto	10		CEP com formato xx.xxx-xxx
logradouro	Simples	Texto	150		Rua, avenida etc.
numero	Simples	Numérico			Número da residencia, edifício etc.
complemento	Simples	Texto	50		Código do bloco, apartamento, casa etc.
bairro	Simples	Texto	50		Nome do bairro
cidade	Simples	Texto	50		Nome da cidade
estado	Simples	Texto	2		Sigla do estado com 2 dígitos

Entidade: categoria					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_categoria	Determinante	Numérico		Serial	
nome	Simples	Texto	255		Nome da categoria

Relacionamento: categoria_subordinada					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_super_categoria	Determinante	Numérico			Código da categoria pai
codigo_sub_categoria	Determinante	Numérico			Código da categoria filho

Entidade: editora					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_editora	Determinante	Numérico		Serial	
nome	Simples	Texto	255		Nome da editora

Entidade: emprestimo					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_emprestimo	Determinante	Numérico		Serial	
data_emprestimo	Simples	Data		Data atual	Data de efetuação do empréstimo
hora_emprestimo	Simples	Hora		Hora atual	Hora de efetuação do empréstimo
data_final	Simples	Data		Data atual + 10	Data final para a devolução
codigo_usuario	Simples	Numérico			Código do usuário atendido
codigo_bibliotecario	Simples	Numérico			Código do bibliotecario que atendeu
codigo_exemplar	Simples	Numérico			Código do exemplar emprestado

Entidade: devolucao					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_devolucao	Determinante	Numérico		Serial	
data_devolucao	Simples	Data		Data atual	Data de efetuação da devolução
hora_devolucao	Simples	Hora		Hora atual	Hora de efetuação da devolução
codigo_emprestimo	Simples	Numérico			Código do empréstimo

Entidade: multa					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_multa	Determinante	Numérico		Serial	
dias_atraso	Simples	Numérico			Número de dias de atraso
valor	Simples	Moeda			Valor total da multa
data_pagamento	Simples	Data		Data atual	Data de efetuação do pagamento
hora_pagamento	Simples	Hora		Hora atual	Hora de efetuação do pagamento
codigo_emprestimo	Simples	Numérico			Código do empréstimo

Entidade: estante					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_estante	Determinante	Numérico		Serial	
Local	Simples	Texto	255		Localização da estante

Entidade: secao					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_secao	Determinante	Numérico		Serial	
nome	Simples	Texto	255		Nome para identificação

Relacionamento: estante_secao					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_secao	Determinante	Numérico			Código da seção
codigo_estante	Determinante	Numérico			Código da estante

Entidade: livro					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
isbn	Determinante	Numérico			Código do livro
data_publicacao	Simples	Data			Data de publicação
titulo	Simples	Texto	255		Título do livro
idioma	Simples	Texto	5		Sigla do idioma, seguindo o padrão xx-xx
codigo_autor	Simples	Numérico			Código do autor
codigo_editora	Simples	Numérico			Código da editora
codigo_categoria	Simples	Numérico			Código da categoria

Entidade: exemplar					
Atributo	**Classe**	**Domínio**	**Tamanho**	**Valor padrão**	**Descrição**
codigo_exemplar	Determinante	Numérico		Serial	
codigo_estante	Simples	Numérico			Código da estante
data_aquisicao	Simples	Data		Data atual	Data de aquisição do exemplar
descricao	Simples	Texto	255		Descrição/observação para registro
isbn	Simples	Numérico			Código do livro

Tabela 6 – Dicionário de Dados

5. Definição dos índices para recuperação de dados

Para a melhoria do tempo de execução das consultas do sistema, foram utilizados os índices B-tree, que são índices padrão do PostgreSQL. Para a definição destes índices, foram analisados os dados únicos para cada entidade, definindo-se então o campo ‘cpf’ das tabelas ‘usuario’ e ‘bibliotecario’, assim como o campo ‘isbn’ da tabela ‘livro’. Durante o trabalho, não foi notada diferença significativa no tempo de execução das consultas com a utilização dos índices, porém, isso provavelmente se deve a pouca quantidade de dados populados no banco, com os resultados devendo ser vistos com uma população maior. A seguir estão os códigos de criação dos índices mencionados:

```
CREATE UNIQUE INDEX IF NOT EXISTS unq_cpf_usu
    ON public.usuario USING btree
    (cpf COLLATE pg_catalog."default" ASC NULLS LAST);

CREATE UNIQUE INDEX IF NOT EXISTS unq_cpf_bib
    ON public.bibliotecario USING btree
    (cpf COLLATE pg_catalog."default" ASC NULLS LAST);

CREATE UNIQUE INDEX IF NOT EXISTS unq_isbn
    ON public.livro USING btree
    (isbn ASC NULLS LAST);
```

6. Elementos de processamento de consulta

Para otimização e desempenho, foi aplicada a ferramenta VACUUM FULL ANALYZE em todas as tabelas do banco. No banco de dados, as tuplas excluídas ou obsoletas por uma atualização não são removidas fisicamente das tabelas, apenas são marcadas como inúteis. Dessa forma, o VACUUM realiza a exclusão definitiva destes dados, enquanto o FULL, reorganiza as tabelas para que os dados não fiquem espaçadas devido as informações antigas. O ANALYZE atualiza as estatísticas usadas pelo planejador para determinar a maneira mais eficiente de executar uma consulta.

Assim como na etapa de criação dos índices, não foi possível perceber os resultados desta função no banco desenvolvido, devido a baixa quantidade de dados populados, além da pouca utilização do banco, onde em um uso como produto final, após diversas inserções, exclusões e atualizações nos dados, apareceriam diversas tuplas inúteis e dados espaçados, que então poderiam ser resolvidos pela ferramenta aplicada.

.

7. Regras contidas no banco

Para o bom funcionamento de um sistema e garantia de integridade das informações, é interessante a existência de regras implementadas no próprio banco de dados. Pensando nisso, foram definidas as seguintes regras no banco construído:

Regras			
Tabela	**Coluna**	**Tipo de regra**	**Valor**
autor	codigo_autor	Valor padrão	Serial
bibliotecario	codigo_bibliotecario	Valor padrão	Serial
usuario	codigo_usuario	Valor padrão	Serial
categoria	codigo_categoria	Valor padrão	Serial
emprestimo	codigo_emprestimo	Valor padrão	Serial
emprestimo	data_emprestimo	Valor padrão	Data atual
emprestimo	hora_emprestimo	Valor padrão	Hora atual
emprestimo	data_final	Valor padrão	Data atual + 10
devolucao	codigo_devolucao	Valor padrão	Serial
devolucao	data_devolucao	Valor padrão	Data atual
devolucao	hora_devolucao	Valor padrão	Hora atual
multa	codigo_multa	Valor padrão	Serial
multa	data_pagamento	Valor padrão	Data atual
multa	hora_pagamento	Valor padrão	Hora atual
estante	codigo_estante	Valor padrão	Serial
secao	codigo_secao	Valor padrão	Serial
exemplar	codigo_exemplar	Valor padrão	Serial
exemplar	data_aquisicao	Valor padrão	Data atual

Tabela 7 – Regras do banco de dados

8. Gatilhos e asserções

Asserções são definidas como condições que devem ser satisfeitas durante as operações em um banco de dados, enquanto gatilhos se referem a execuções que devem ser feitas de forma automática após alguma modificação no banco. Neste projeto, caso ocorra atraso em uma devolução, deve ser cobrada uma multa do usuário, com base no período de atraso. Pensando nisso, foi criada uma asserção para garantir que um usuário que esteja com alguma pendência não possa realizar novos empréstimos, enquanto não a regularizar. O código para executar esta asserção como consulta é o seguinte:

```
DO $$
DECLARE
   atraso_count integer;
BEGIN
   SELECT COUNT(*) INTO atraso_count
      FROM ((SELECT e.codigo_usuario
            FROM emprestimo e
            FULL JOIN devolucao d ON d.codigo_emprestimo =
e.codigo_emprestimo
            WHERE e.data_emprestimo = CURRENT_DATE)
         INTERSECT
         (SELECT e.codigo_usuario
            FROM emprestimo e
            FULL JOIN devolucao d ON d.codigo_emprestimo =
e.codigo_emprestimo
            WHERE e.data_final < CURRENT_DATE
            AND d.data_devolucao is NULL)) AS foo;
   assert atraso_count = 0, 'Atraso pendente identificado.';
end$$
```

Para implementar esta asserção de forma que seja verificada toda vez que um empréstimo novo é cadastrado, impedindo-o caso seja identificada alguma pendência, foi utilizado um trigger, com o seguinte código:

```
CREATE OR REPLACE FUNCTION checkEmprestimoPendencia()
   RETURNS TRIGGER AS $$BEGIN IF
      (SELECT COUNT(*) AS pendencia_count
         FROM ((SELECT e.codigo_usuario
            FROM emprestimo e
            FULL JOIN devolucao d ON d.codigo_emprestimo =
e.codigo_emprestimo
            WHERE e.data_emprestimo = CURRENT_DATE)
         INTERSECT
         (SELECT e.codigo_usuario
            FROM emprestimo e
            FULL JOIN devolucao d ON d.codigo_emprestimo =
e.codigo_emprestimo
            WHERE e.data_final < CURRENT_DATE
```

```
                    AND d.data_devolucao is NULL)) AS foo) > 0
  THEN RAISE EXCEPTION 'Atraso pendente identificado.';
  END IF; RETURN NEW; END$$
  LANGUAGE plpgsql;

CREATE TRIGGER tg_checkEmprestimoPendencia
        AFTER INSERT ON emprestimo
        EXECUTE PROCEDURE checkEmprestimoPendencia();
```

Outra necessidade do banco trabalhado é de, ao ser efetuada a exclusão de um livro, deve-se também ser feita aexclusão dos seus exemplares, afinal o exemplar precisa do livro como referência, além de remover o valor do exemplar na tabela de empréstimos. O mesmo deve ocorrer em relação a exclusão de uma categoria, onde as suas relações de subordinação também devem ser excluídas. Estas operações foram implementadas por meio dos seguintes triggers:

```
CREATE OR REPLACE FUNCTION deleteLivroExemplar()
RETURNS TRIGGER AS $BODY$
BEGIN
   DELETE FROM exemplar
   WHERE codigo_exemplar IN
      (SELECT codigo_exemplar
       FROM exemplar
       WHERE isbn = OLD.isbn);
   RETURN NULL;
END;
$BODY$
LANGUAGE 'plpgsql';

CREATE TRIGGER tg_deleteLivroExemplar
        BEFORE DELETE ON livro FOR EACH ROW EXECUTE PROCEDURE
        deleteLivroExemplar();


CREATE OR REPLACE FUNCTION setNullEmprestimoExemplar()
RETURNS TRIGGER AS $BODY$
BEGIN
   UPDATE emprestimo
   SET codigo_exemplar = NULL
   WHERE codigo_exemplar IN
      (SELECT codigo_exemplar
       FROM exemplar
       WHERE codigo_exemplar = OLD.codigo_exemplar);
   RETURN NULL;
END;
$BODY$
LANGUAGE 'plpgsql';

CREATE TRIGGER tg_setNullExemplar
```

```
        BEFORE DELETE ON livro FOR EACH ROW EXECUTE PROCEDURE
        setNullEmprestimoExemplar();


CREATE OR REPLACE FUNCTION deleteCategoriaSubordinada()
RETURNS TRIGGER AS $BODY$
BEGIN
   DELETE FROM categoria_subordinada
   WHERE codigo_super_categoria = OLD.codigo_categoria
   OR codigo_sub_categoria = OLD.codigo_categoria;
   RETURN OLD;
END;
$BODY$
LANGUAGE 'plpgsql';

CREATE TRIGGER tg_deleteCategoriaSubordinada
        BEFORE DELETE ON categoria FOR EACH ROW EXECUTE
        PROCEDURE setNullEmprestimoExemplar();
```

9. Conjunto de SQLs

Após o desenvolvimento e população do banco de dados, foram elaboradas 10 consultas, descrevendo os seus enunciados e analisando-se as suas complexidades com notas de 0 a 10, onde abaixo de 6 o grau de complexidade de elaboração é considerado baixo, de 6 a 8 é médio e acima de 8 é alto, conforme a seguir:

- **Consulta 1:** Consultar o nome, rg e cidade do endereço principal (primeiro endereço) de todos os usuários da biblioteca e, caso possua multas cadastradas com 3 ou mais dias de atraso, mostrar os dias de atraso para cada uma, o nome do livro emprestado, o valor da multa e a data de pagamento, com os dados mostrados em ordem alfabética pelo nome da cidade. Caso não possua multas, deverá exibir a frase "Não possui" nos campos referentes a multa.

  Funcionamento: A consulta envolve as tabelas usuario, endereco_usuario, multa, emprestimo, exemplar e livro, com WITH AS para a sub consulta dos usuários com multa, INNER JOIN e LEFT JOIN para as junções e uso de COALESCE em dados para quando não existam multas para o usuário, além de ORDER BY para a ordenação.

Código:

```
WITH um AS
   (SELECT u.codigo_usuario, m.dias_atraso, l.titulo, m.valor,
   m.data_pagamento
   FROM usuario u
   JOIN emprestimo e ON e.codigo_usuario = u.codigo_usuario
   JOIN multa m ON m.codigo_emprestimo = e.codigo_emprestimo AND
   m.dias_atraso >= 3
   LEFT JOIN devolucao d ON d.codigo_emprestimo = e.codigo_emprestimo
   LEFT JOIN exemplar ex ON ex.codigo_exemplar = e.codigo_exemplar
   LEFT JOIN livro l ON l.isbn = ex.isbn
   WHERE m.dias_atraso >= 3)

SELECT u.nome, u.rg, eu.cidade, COALESCE(um.dias_atraso::text, 'Não
possui') AS dias_atraso, COALESCE(um.titulo, 'Não possui') AS titulo,
COALESCE(um.valor::text, 'Não possui') AS valor,
COALESCE(um.data_pagamento::text, 'Não possui') AS data_pagamento
FROM usuario u
JOIN endereco_usuario eu ON eu.codigo_usuario = u.codigo_usuario AND
codigo_endereco_usuario = 1
LEFT JOIN um ON um.codigo_usuario = u.codigo_usuario
ORDER BY eu.cidade;
```

Avaliação: Devido a necessidade de uma sub consulta com o comando WITH AS, além de varios JOINs e alterações de dados, esta consulta foi avaliada com uma nota de 7, com um grau médio de complexidade de elaboração.

- **Consulta 2:** Consultar o nome, rg, telefone e cidade do endereço principal (primeiro endereço) de todos os bibliotecários, assim como o número de empréstimos que já efetuaram e de devoluções e multas que já receberam, com os valores mostrados em ordem decrescente pelo número de empréstimos efetuados.

  Funcionamento: A consulta envolve as tabelas bibliotecario, endereco_bibliotecario, multa, emprestimo e devolucao, com WITH AS para as sub consultas, INNER JOIN e LEFT JOIN para as junções, GROUP BY e COUNT para os totais, COALESE para dados nulos e ORDER BY para a ordenação.

  Código:

```
WITH be AS
   (SELECT e.codigo_bibliotecario, count(e.codigo_bibliotecario)
    FROM bibliotecario b
    JOIN emprestimo e ON e.codigo_bibliotecario = b.codigo_bibliotecario
    GROUP BY (e.codigo_bibliotecario)
   ),
bm AS
   (SELECT m.codigo_bibliotecario, count(m.codigo_bibliotecario)
    FROM bibliotecario b
    JOIN emprestimo e ON e.codigo_bibliotecario = b.codigo_bibliotecario
    JOIN multa m ON m.codigo_emprestimo = e.codigo_emprestimo
    GROUP BY (m.codigo_bibliotecario)
   ),
bd AS
   (SELECT d.codigo_bibliotecario, count(d.codigo_bibliotecario)
    FROM bibliotecario b
    JOIN emprestimo e ON e.codigo_bibliotecario = b.codigo_bibliotecario
    JOIN devolucao d ON d.codigo_emprestimo = e.codigo_emprestimo
    GROUP BY (d.codigo_bibliotecario)
   )
```

```
SELECT b.nome, b.rg, b.telefone, eb.cidade, COALESCE(be.count, 0) AS
emprestimos,
COALESCE(bd.count, 0) AS devolucoes, COALESCE(bm.count, 0) AS multas
FROM bibliotecario b
JOIN endereco_bibliotecario eb ON eb.codigo_bibliotecario =
b.codigo_bibliotecario AND eb.codigo_endereco_bibliotecario = 1
JOIN be ON be.codigo_bibliotecario = b.codigo_bibliotecario
LEFT JOIN bm ON bm.codigo_bibliotecario = b.codigo_bibliotecario
LEFT JOIN bd ON bd.codigo_bibliotecario = b.codigo_bibliotecario
ORDER BY emprestimos DESC;
```

Avaliação: Esta consulta utiliza 3 subconsultas com WITH AS, além de GROUP BY e COUNT para a contagem de dados, sendo então um pouco mais complexa que a consulta 1. Portando, esta consulta foi avaliada com uma nota 7,5, com um grau médio de complexidade de elaboração.

- **Consulta 3:** Consultar o título e nome do autor de cada livro, assim como o número total de exemplares disponíveis e o total de empréstimos realizados, com os valores mostrados em ordem decrescente pelo número de exemplares.

  Funcionamento: A consulta envolve as tabelas livro, exemplar, autor e empréstimo, com uso de INNER JOIN e LEFT JOIN para as junções, GROUP BY para obter os totais e COALESE para o caso de falta de empréstimos ou exemplares.

  Código:

```
WITH tex AS
   (SELECT e.isbn, COUNT(e.isbn)
    FROM exemplar e
    GROUP BY (e.isbn)),
tem AS
   (SELECT ex.isbn, COUNT(ex.isbn)
    FROM exemplar ex
    JOIN emprestimo em ON em.codigo_exemplar = ex.codigo_exemplar
    GROUP BY (ex.isbn))

SELECT l.titulo, a.nome, COALESCE(tex.count, 0) AS total_exemplares,
COALESCE(tem.count, 0) AS total_emprestimos
FROM livro l
```

```
JOIN autor a ON a.codigo_autor = l.codigo_autor
LEFT JOIN tex ON tex.isbn = l.isbn
LEFT JOIN tem ON tem.isbn = l.isbn
ORDER BY total_exemplares DESC;
```

Avaliação: Esta consulta, apesar de utilizar duas sub consultas com GROUP BY e COUNT, é menor do que as anteriores, utilizando menos junções e comparações, portanto a sua nota ficou como 6, com um grau médio de complexidade de elaboração.

- **Consulta 4:** Encontrar o nome de todos os livros e, para cada um, mostrar a data de empréstimo de seu exemplar que resultou na multa de maior valor de atraso, assim como o nome do usuário que efetuou o empréstimo e o valor da multa, em ordem alfabética pelo nome do usuário. Caso não haja multas para o livro, exibir a frase "Sem multas" nos campos referente a multa.

  Funcionamento: A consulta envolve as tabelas livro, exemplar, emprestimo, multa e usuario, envolvendo o uso de INNER JOIN e LEFT JOIN, além das funções WITH() e MAX() para a obtenção dos maiores valores e ORDER BY para a ordenação.

  Código:

```
WITH mv AS
   (SELECT l.isbn, MAX(m.valor) AS valor
   FROM livro l
   JOIN exemplar ex ON ex.isbn = l.isbn
   JOIN emprestimo em ON em.codigo_exemplar = ex.codigo_exemplar
   JOIN multa m ON m.codigo_emprestimo = em.codigo_emprestimo
   GROUP BY (l.isbn)),
emv AS
   (SELECT l.isbn, em.data_emprestimo, u.nome, m.valor
   FROM livro l
   LEFT JOIN mv ON mv.isbn = l.isbn
   LEFT JOIN exemplar ex ON ex.isbn = l.isbn
   LEFT JOIN emprestimo em ON em.codigo_exemplar = ex.codigo_exemplar
   LEFT JOIN multa m ON m.codigo_emprestimo = em.codigo_emprestimo
   LEFT JOIN usuario u ON u.codigo_usuario = em.codigo_usuario
   WHERE m.valor=mv.valor)

SELECT l.titulo, COALESCE(emv.data_emprestimo::text, 'Sem multas') AS
data_emprestimo,
COALESCE(emv.nome, 'Sem multas') AS nome_usuario,
COALESCE(emv.valor::text, 'Sem multas') AS valor_multa
FROM livro l
LEFT JOIN emv ON emv.isbn = l.isbn
ORDER BY nome_usuario;
```

Avaliação: Esta consulta é um pouco mais complexa do que a anterior, envolvendo a função MAX e duas sub consultas, porém é menos complexa do que as demais, portanto a sua nota ficou como 6,5, com um grau médio de complexidade de elaboração

- **Consulta 5:** Exibir o número total de emprestimos por usuário, assim como o número total de multas geradas e de devoluções recebidas, em ordem alfabética pelo nome do usuario. Caso não exista algum valor para o usuário, exibir a frase “Sem dados”.

  Funcionamento: A consulta envolve as tabelas exemplar, livro, emprestimo, multa, devolucao e usuario, com uso de INNER JOIN e LEFT JOIN para as junções, WITH() para as sub consultas, COUNT() e GROUP BY para a obtenção dos valores totais e de ORDER BY para a ordenação.

  Código:

```
WITH te AS
   (SELECT codigo_usuario, COUNT(codigo_usuario)
    FROM emprestimo
    GROUP BY(codigo_usuario)
   ),
tm AS
   (SELECT e.codigo_usuario, COUNT(e.codigo_usuario)
    FROM emprestimo e
    JOIN multa m ON m.codigo_emprestimo = e.codigo_emprestimo
    GROUP BY(codigo_usuario)
   ),
td AS
   (SELECT e.codigo_usuario, COUNT(e.codigo_usuario)
    FROM emprestimo e
    JOIN devolucao d ON d.codigo_emprestimo = e.codigo_emprestimo
    GROUP BY(codigo_usuario)
   )

SELECT u.nome, COALESCE(te.count::text, 'Sem dados') AS emprestimos,
COALESCE(tm.count::text, 'Sem dados') AS multas,
COALESCE(td.count::text, 'Sem dados') AS devolucoes
FROM usuario u
LEFT JOIN te ON te.codigo_usuario = u.codigo_usuario
LEFT JOIN tm ON tm.codigo_usuario = u.codigo_usuario
LEFT JOIN td ON td.codigo_usuario = u.codigo_usuario;
```

  Avaliação: Esta consulta possui 3 sub consultas, porém pouco complexas, com a consulta principal demandando alguns JOINs, de tamanho semelhante à consulta 3, porém um pouco maior, então a sua nota ficou como 6,3, com um grau médio de complexidade de elaboração.

- **Consulta 6:** Encontrar a categoria onde ocorreu o maior número de empréstimos, mostrando o número total de empréstimos e de multas geradas, além da média de dias de atraso.

  Funcionamento: A consulta envolve as tabelas categoria, emprestimo e multa, com uso de INNER JOIN e LEFT JOIN para as junções, além de COALESE para dados nulos e WITH(), COUNT() GROUP BY e MAX() para a obtenção do maior valor e dos números totais e AVG() para as médias.

  Código:

```
WITH tec AS
   (SELECT c.codigo_categoria, c.nome, COUNT(c.codigo_categoria)
   FROM emprestimo em
   JOIN exemplar ex ON ex.codigo_exemplar = em.codigo_exemplar
   JOIN livro l ON ex.isbn = l.isbn
   JOIN categoria c ON c.codigo_categoria = l.codigo_categoria
   GROUP BY (c.codigo_categoria, c.nome)),
tmc AS
   (SELECT c.codigo_categoria, AVG(m.dias_atraso) AS media_atraso,
   COUNT(c.codigo_categoria) AS total_multas
   FROM emprestimo em
   JOIN multa m ON m.codigo_emprestimo = em.codigo_emprestimo
   JOIN exemplar ex ON ex.codigo_exemplar = em.codigo_exemplar
   JOIN livro l ON ex.isbn = l.isbn
   JOIN categoria c ON c.codigo_categoria = l.codigo_categoria
   JOIN tec on tec.codigo_categoria = c.codigo_categoria
   GROUP BY (c.codigo_categoria))

SELECT tec.nome, tec.count AS total_emprestimos,
COALESCE(tmc.total_multas::text, 'Sem multas') AS total_multas,
COALESCE(tmc.media_atraso::text, 'Sem multas') AS media_atraso
FROM tec
LEFT JOIN tmc ON tmc.codigo_categoria = tec.codigo_categoria
WHERE tec.codigo_categoria IN
   (SELECT codigo_categoria FROM tec WHERE count IN
      (SELECT MAX(count) AS codigo_categoria FROM tec));
```

  Avaliação: Esta consulta, além de demandar 2 sub consultas com diversos JOINs, utiliza também as funções AVG e 2 consultas aninhadas na consulta principal, tendo um nível de complexidade semelhante a consulta 2. Portanto, a sua nota ficou como 7,5, com um grau de complexidade mério de elaboração.

- **Consulta 7:** Encontrar o usuario com o maior total de dias de atraso (soma de atraso de todos os emprestimos realizados) e mostrar também o valor total pago em multas.

  Funcionamento: A consulta envolve as tabelas usuario, emprestimo e multa, com o uso de INNER JOIN e das funções WITH(), MAX(), COUNT(), SUM() e GROUP BY para a obtenção dos valores totais e maior valor.

  Código:

```
WITH um AS
   (SELECT u.codigo_usuario, m.dias_atraso, m.valor
   FROM emprestimo e
   JOIN usuario u ON u.codigo_usuario = e.codigo_usuario
   JOIN multa m ON m.codigo_emprestimo = e.codigo_emprestimo),
umt AS
   (SELECT u.codigo_usuario, SUM(um.dias_atraso) AS total_dias_atraso,
SUM(um.valor) AS total_valor_pago
   FROM usuario u
   JOIN um ON u.codigo_usuario = um.codigo_usuario
   GROUP BY (u.codigo_usuario))

SELECT u.nome, umt.total_dias_atraso, umt.total_valor_pago
FROM usuario u
JOIN umt ON umt.codigo_usuario = u.codigo_usuario
WHERE umt.total_dias_atraso IN
   (SELECT MAX(total_dias_atraso) FROM umt);
```

  Avaliação: Esta consulta utiliza recursos semelhantes à consulta 3, porém com o uso de MAX e uma consulta aninhada na consulta principal. Portanto, a sua nota ficou como 6,3, com um nível médio de complexidade de elaboração.

- **Consulta 8:** Encontrar o nome do bibliotecário da cidade de Cascavel que efetuou o maior número de empréstimos no ano de 2022, exibindo também o número total de empréstimos efetuados por ele.

  Funcionamento: A consulta envolve as tabelas bibliotecario, endereco_bibliotecario e emprestimo, com uso de INNER JOIN e das funções WITH(), COUNT(), MAX() e GROUP BY para a obtenção dos valores totais e do maior valor, com 4 condições de igualdade em cláusula WHERE.

  Código:

```
WITH emb AS
   (SELECT b.codigo_bibliotecario, COUNT(b.codigo_bibliotecario) AS
emprestimos
   FROM emprestimo e
   JOIN bibliotecario b ON b.codigo_bibliotecario = e.codigo_bibliotecario
   JOIN endereco_bibliotecario eb ON eb.codigo_bibliotecario =
b.codigo_bibliotecario
```

```
    WHERE eb.codigo_endereco_bibliotecario = 1
    AND eb.cidade = 'Cascavel'
    AND DATE_PART('year', e.data_emprestimo) = 2022
    GROUP BY (b.codigo_bibliotecario))

SELECT b.nome, emb.emprestimos
FROM bibliotecario b
JOIN emb ON emb.codigo_bibliotecario = b.codigo_bibliotecario
WHERE emb.emprestimos IN
    (SELECT MAX(emprestimos) FROM emb);
```

Avaliação: Esta consulta utiliza 3 sub consultas, MAX e uma seleção aninhada na seleção principal, sendo semelhante à consulta 7 em termos de complexidade. Portanto, a sua nota ficou como 6,3, com um grau médio de complexidade de elaboração.

- **Consulta 9:** Encontrar o nome dos usuários da cidade de Toledo que efetuaram devoluções em janeiro de 2022 sem atraso/multa, assim como o número total de devoluções que efetuaram nesse período.

  Funcionamento: A consulta envolve as tabelas usuario, endereco_usuario, emprestimo, devolucao e multa, com uso de INNER JOIN e das funções WITH(), COUNT() e GROUP BY para a obtenção dos valores totais e 4 condições em cláusula WHERE.

  Código:

```
WITH du AS
    (SELECT u.codigo_usuario, COUNT(u.codigo_usuario)
    FROM emprestimo em
    JOIN usuario u ON u.codigo_usuario = em.codigo_usuario
    JOIN devolucao d ON d.codigo_emprestimo = em.codigo_emprestimo
    JOIN endereco_usuario eu ON eu.codigo_usuario = u.codigo_usuario
    LEFT JOIN multa m ON m.codigo_emprestimo = em.codigo_emprestimo
    WHERE eu.codigo_endereco_usuario = 1
    AND eu.cidade = 'Toledo'
    AND DATE_PART('month', d.data_devolucao) = 01
    AND m.codigo_multa IS NULL
    GROUP BY (u.codigo_usuario))

SELECT u.nome, du.count AS devolucoes
FROM usuario u
JOIN du ON du.codigo_usuario = u.codigo_usuario;
```

  Avaliação: Esta consulta utiliza apenas uma subconsulta, com alguns JOINs e comparações na cláusula WHERE, tendo então uma complexidade semelhante a consulta 3. Portanto, a sua nota ficou como 6, com um grau médio de complexidade de elaboração.

- **Consulta 10:** Encontrar o nome do autor cadastrado que possui a maior quantidade de livros publicados entre os anos 2000 e 2005, assim como o número total de categorias em que publicou neste período.

  Funcionamento: A consulta envolve as tabelas autor, livro e categoria, com uso de INNER JOIN e das funções WITH(), COUNT(), GROUP BY e MAX() para a obtenção dos valores totais e maior valor, além de 3 condições de desigualdade na clausula WHERE.

  Código:

```
WITH lp AS
   (SELECT a.codigo_autor, COUNT(a.codigo_autor)
   FROM livro l
   JOIN autor a ON a.codigo_autor = l.codigo_autor
   JOIN categoria c ON c.codigo_categoria = l.codigo_categoria
   WHERE DATE_PART('year', l.data_publicacao) >= 2000
   AND DATE_PART('year', l.data_publicacao) <= 2005
   GROUP BY(a.codigo_autor))

SELECT a.nome, lp.count
FROM autor a
JOIN lp ON lp.codigo_autor = a.codigo_autor
WHERE lp.count IN
   (SELECT MAX(count) FROM lp)
```

  Avaliação: Esta consulta possui apenas um with, com poucos joins e validações, apesar de uma seleção aninhada na seleção principal e uso de MAX, possuindo então uma complexidade semelhante à da consulta 9. Portanto, a sua nota ficou como 6, com um grau médio de complexidade de elaboração.

10. Considerações Finais

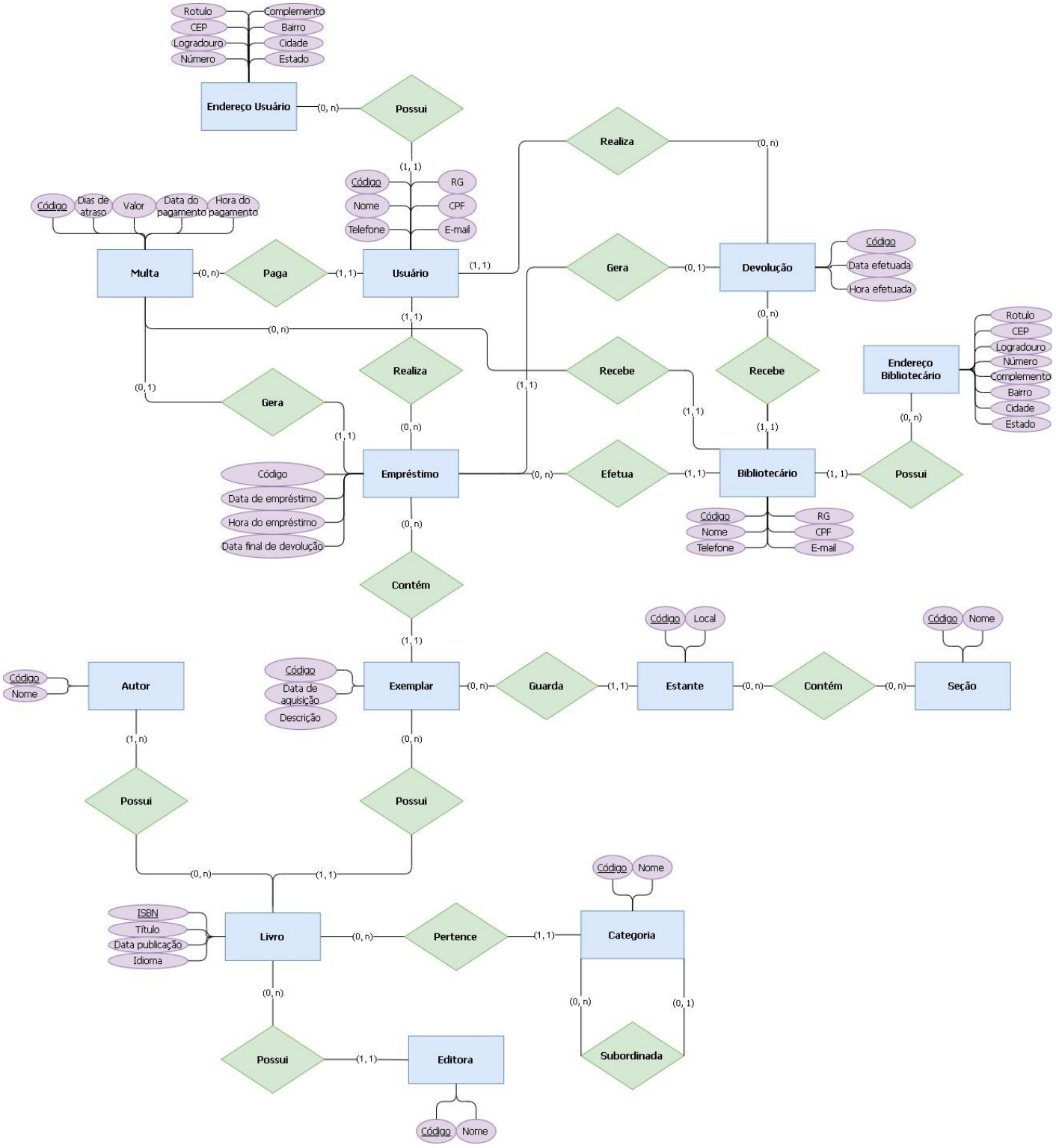
Este trabalho permitiu o exercício do planejamento de um sistema de banco de dados, com o levantamento de requisitos, elaboração dos diagramas, idealização do dicionário de dados até chegar-se a etapa final de desenvolvimento. Com isso, foi possível aplicar de forma prática diversos pontos da disciplina vistos em aula, o que permitiu à equipe sanar dúvidas e se familiarizar mais com os processos de planejamento e desenvolvimento do banco, gerando um aprofundamento ao conteúdo estudado durante o ano letivo.

As otimizações aplicadas no banco, com a modelagem cumprindo as formas normais, assim como a utilização de índices e os elementos de processamento de consulta foram etapas importantes para um bom funcionamento entre as tabelas e

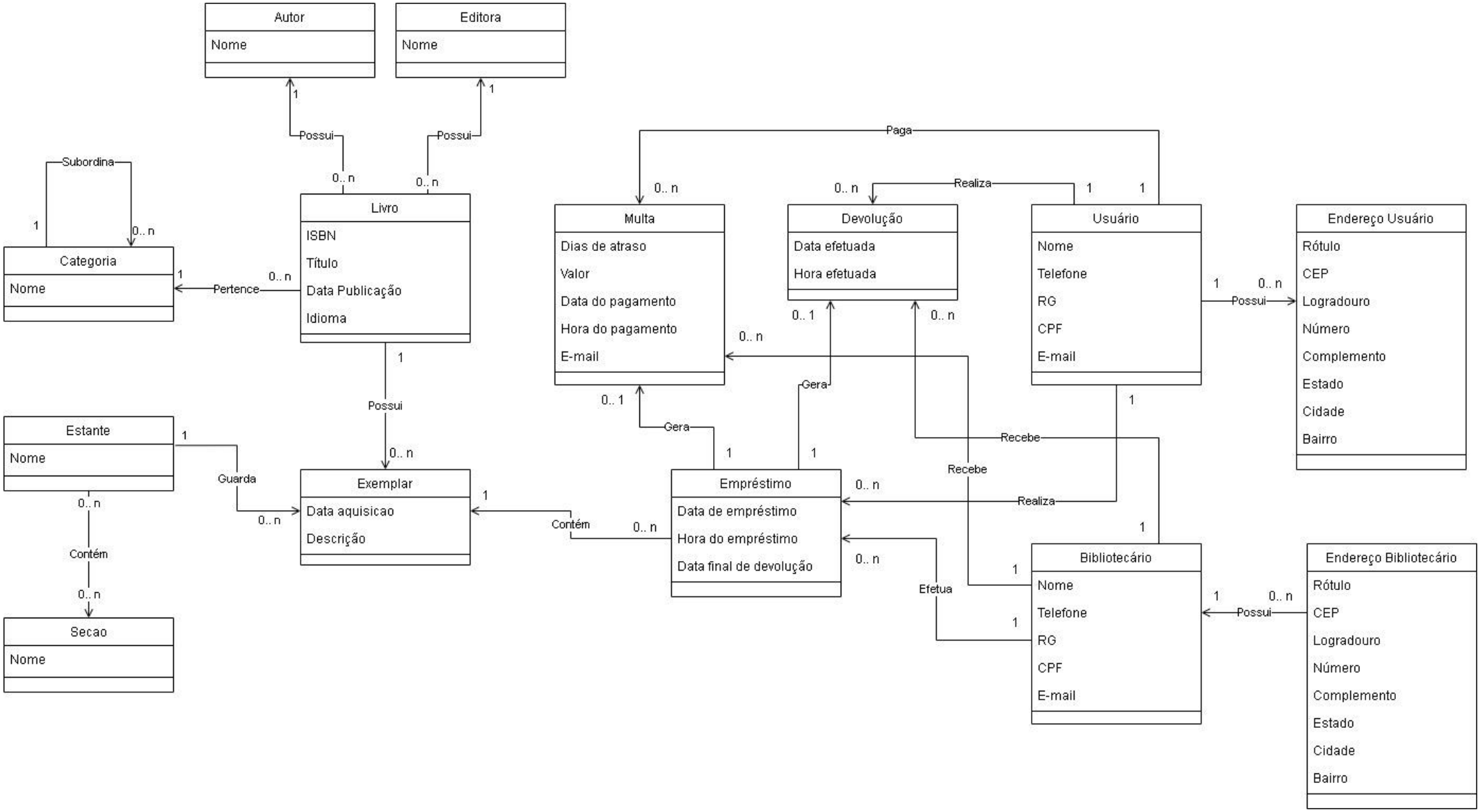
otimização do banco, onde com uma quantidade maior de dados populados e maior utilização, seria possível visualizar um ganho de desempenho nas consultas.

A configuração de regras, gatilhos, asserções permitiu que vários dados sejam validados no proprio banco, garantindo uma maior integridade para as informações armazenadas, evitando que valores indesejados sejam processados.

ANEXO I - DIAGRAMA ENTIDADE RELACIONAMENTO



ANEXO II – DIAGRAMA DE CLASSES



ANEXO III – MODELO RELACIONAL

Autor	
PK	**codigo_autor**
	nome

Editora	
PK	**codigo_editora**
	nome

Categoria	
PK	**codigo_categoria**
	nome

Categoria_subordinada	
PK,FK2	**codigo_sub_categoria**
PK,FK1	**codigo_super_categoria**

Estante	
PK	**codigo_estante**
	Local

Secao	
PK	**codigo_secao**
	nome

Estante_secao	
PK,FK2	**codigo_estante**
PK,FK1	**codigo_secao**

Bibliotecario	
PK	**codigo_bibliotecario**
	nome
	telefone
	nascimento
	rg
	cpf
	email

Endereco_bibliotecario	
PK	codigo_endereco_bibliotecario
PK,FK	**codigo_bibliotecario**
	rotulo
	cep
	logradouro
	numero
	complemento
	bairro
	cidade
	estado

Usuario	
PK	**codigo_usuario**
	nome
	telefone
	nascimento
	rg
	cpf
	email

Endereco_usuario	
PK	codigo_endereco_usuario
PK,FK	**codigo_usuario**
	rotulo
	cep
	logradouro
	numero
	complemento
	bairro
	cidade
	estado

Emprestimo	
PK	**codigo_emprestimo**
FK1	codigo_usuario
FK2	codigo_bibliotecario
FK3	codigo_exemplar
	data_emprestimo
	hora_emprestimo
	data_final

Devolucao	
PK	**codigo_devolucao**
FK	codigo_emprestimo
FK	codigo_bibliotecario
	data_devolucao
	hora_devolucao

Multa	
PK	**codigo_multa**
FK	codigo_emprestimo
FK	codigo_bibliotecario
	dias_atraso
	valor
	data_pagamento
	hora_pagamento

Livro	
PK	**isbn**
PK,FK1	codigo_editora
PK,FK2	codigo_categoria
PK,FK3	codigo_estante
	data_publicacao
	titulo
	idioma

Exemplar	
PK	**codigo_exemplar**
PK,FK	isbn
	data_aquisicao
	descricao