# 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

# 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.

# 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*

# 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 ilustra como 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.

# 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*

# 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);

# 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.

.

# 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*

# 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();

# 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.

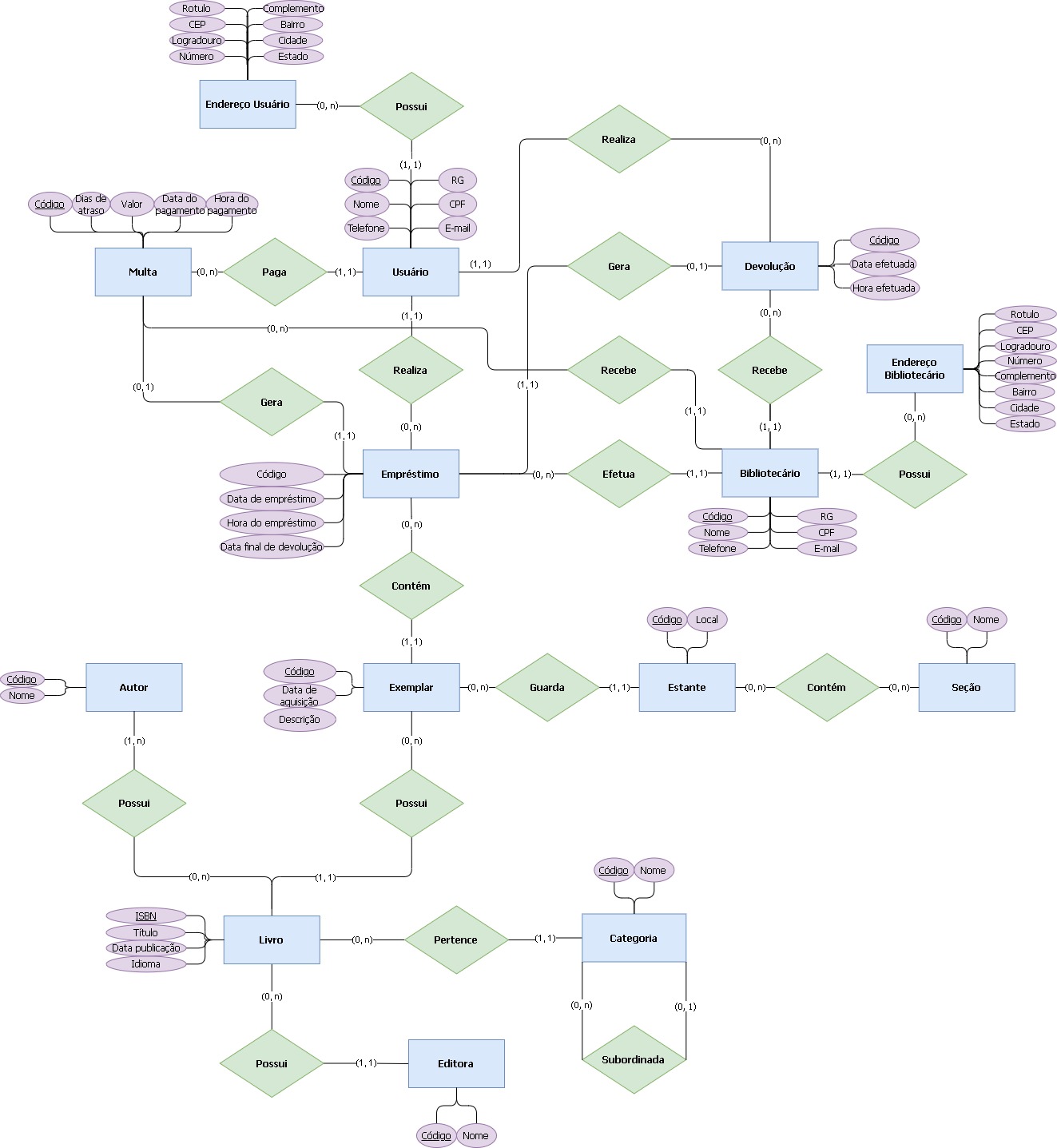
# Considerações Finais

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.

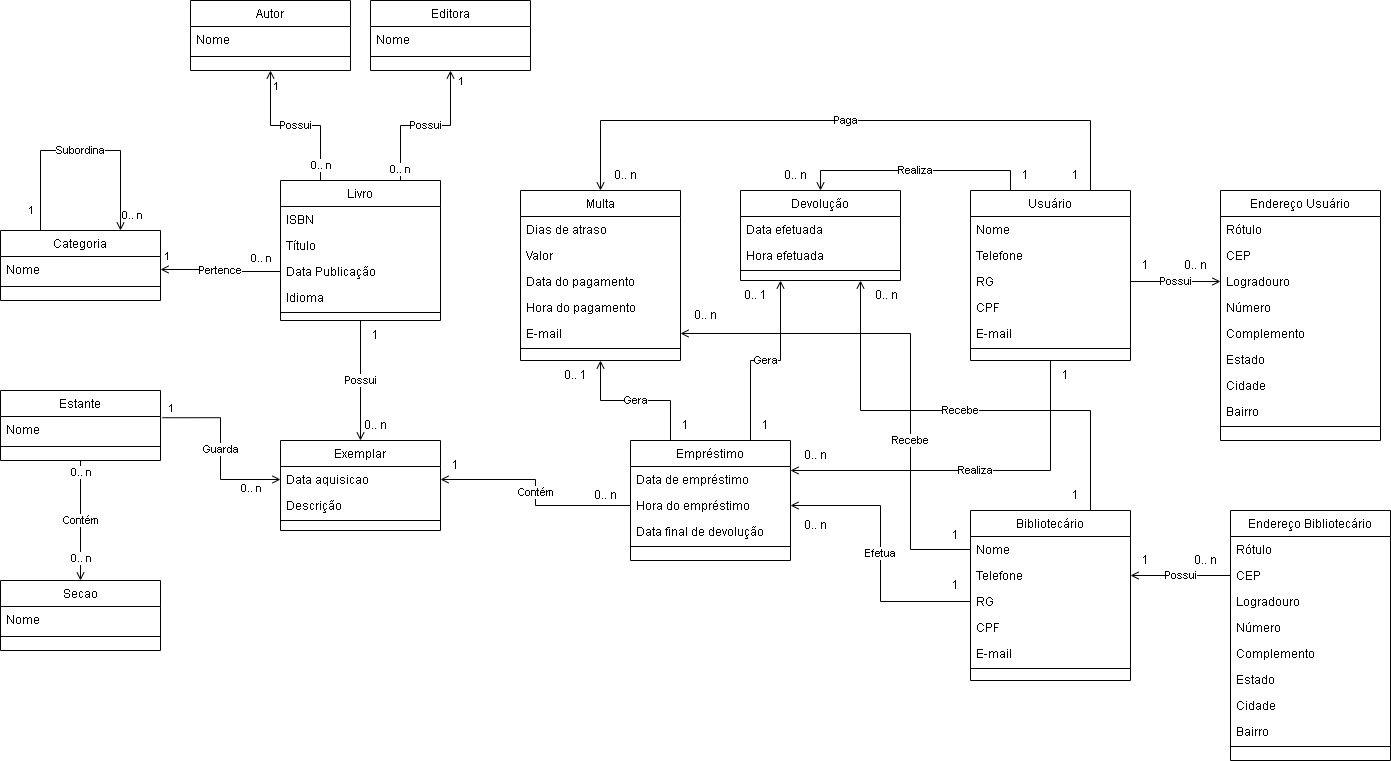
As otimizações aplicadas no banco, com a modelagem cumprindo as formas normais, assim como a utilização de índices e os elementos de processamento de consulta foram etapas importantes para um bom funcionamento entre as tabelas e otimização do banco, onde com uma quantidade maior de dados populados e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

