Licenciatura Engenharia Informática

Ciência de Dados

Base de Dados MSSQL
2 Semestre

Executado por
45206 Bruno Martins
30156 Pedro Rodrigues
40204 Pedro Pinto

Orientado por
Ricardo Ferreira

Entregue em
20/04/2022

1. Resumo

Em ambiente de produção, de modo a garantir a fiabilidade, a segurança e a consistência da base de dados, é necessário que o esquema da base de dados esteja em conformidade com as formas de normalização dos dados: **1FN, 2FN, 3FN (4FN e 5FN)** para contextos mais específicos.

Com as regras de 1:1, 1:N, N:1, N:N entre outras ligações, permite-se, portanto, a criação de uma base de dados consistente sem perda de informação e com um aspecto visual mais elegante e um comportamento funcional e otimizado.

Com base nestes conhecimentos adquiridos, foi tomado em conta, com estas regras, e, em conformidade com o exercício proposto, criar não apenas uma base de dados para completar o exercício, mas para tentar elevar a base de dados a um nível que possa ser aceite em produção, no mundo real.

Aproveitou-se, portanto, esta oportunidade, para consolidar conhecimentos e evoluir para estes parâmetros acima descritos.

2. Índice

3. Introdução (ou Objectivos)

Para a elaboração deste projeto/trabalho, o tema proposto foi a criação de uma base de dados relacional, simulando diversas situações presentes num banco, como por exemplo, o processo de crédito e associação de uma ou várias contas bancárias a um cliente.

Durante o relatório, será explicado o modelo de entidade e associação, onde será explicado toda a lógica de negócio e de dados utilizados para a resolução do trabalho em questão.

Passando para o DEA, será apresentado conteúdo das entidades, onde será explicado os atributos de cada tabela, de forma a ser compreendido e justificado, algumas das ideias retiradas de aplicações reais, nomeadamente, ligadas ao setor da banca. Na aba do Modelo Relacional, será apresentado uma tabela com as chaves para representar a explicação feita no DEA.

4. Desenvolvimento

4.1 Modelo Entidade Associação - EA

Conforme indicado na introdução inicial deste projeto, para pensar e formatar toda a lógica e ideias das entidades, foi feito um estudo da arte com base em algumas aplicações existentes, onde se observou a organização, estrutura e divisão dos dados, bem como a pesquisa de algum fundamento teórico sobre o tema em questão.

Importante referir que, durante a explicação da lógica das relações e indicação das entidades, foi utilizado um padrão na criação de atributos simbolizado por **uuid**.

O objetivo deste campo é possibilitar e mostrar a identificação do registo de forma humanizada, sem dar “brechas” de segurança, comparado com um simples id do mesmo.

Vamos então iniciar a explicação das entidades por grupos/contextos, para melhor aproveitamento.

(Nos anexos, está disponível a visualização direta dos modelos: entidade associação e relacional.)

4.1.1 Entidade User com relação para *Customer* e *Colaborator*

Nesta associação, considerou-se os seguintes problemas:

- O utilizador, poderá ser um cliente e um colaborador ao mesmo tempo?
- Este, de alguma forma, sendo um cliente, poderá se tornar um colaborador **mais tarde?** (vice versa)

Com base neste pequeno problema, foi criado uma relação de um para um (1..1) de forma a que a tabela *User* contenha todos os dados pessoais, de identificação de forma única, e que, seja utilizado, tanto na entidade *Customer* e *Collaborator* para evitar a replicação de dados sensíveis.

Além desta justificação, as entidades *Customer* e *Collaborator* possuem campos exclusivos para o seu cargo. De forma a evitar chaves nulas nos registos, justifica-se, portanto, a criação destas duas tabelas à parte.

Tendo em conta aos seus atributos:

- **User**
    - id_user, uuid_user, address, dob, phone_number, city, first_name, last_name, cc_identification, nif_identification, personal_email, is_active,created_at, updated_at
- **Customer**
    - id_customer, id_user, contract_number, pin_code, is_active, created_at, updated_at
- **Collaborator**
    - id_collaborator, id_user , work_phone_number, work_email_address, created_at, updated_at

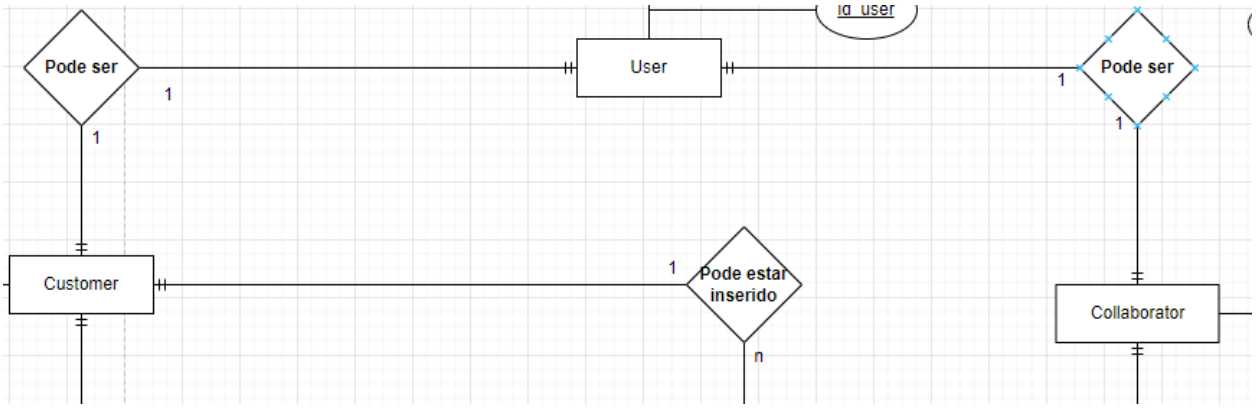


Figura 1: Ligação de **1..1** entre User para as tabelas ***Customer*** e ***User***.

4.1.2 Entidade Conta com a lógica de Titulares e Permissões

Nestas associações, considera-se que um cliente pode estar presente em várias contas e várias contas, podendo, portanto, possuir vários clientes.

Estamos perante uma associação de muitos para muitos, (N..N), e, portanto, é criada uma terceira tabela a juntar as duas principais, designa de Holder.

- De Customer para Holder - **um para muitos**.
- De Holder para Account- **muitos para um**.

Um Titular (Holder), pertence apenas a um Cliente enquanto que o mesmo Cliente pode ser Titular de várias contas.

A mesma lógica é aplicada para a associação entre Holder e Conta.
Um titular está associado a uma conta, enquanto que uma conta poderá ter vários titulares associados.

Com relação a permissões, considerou-se que as permissões deverão ser associadas aos titulares diretamente e não à conta. Caso associado a uma conta, retirava a funcionalidade de poder desativar uma permissão específica a qualquer titular por qualquer motivo.

Portanto, um titular poderá ter nenhuma ou várias permissões e várias permissões serão delegadas aos mesmos.

- De Holder para HolderPermissons - **um para nenhum ou muitos**.
- De HolderPermission para Permissions - **muitos para um**.

Tendo em conta aos seus atributos:

- **Holder**
    - id_holder, id_account, id_customer, register_date, is_active, created_at, updated_at

- **HolderPermission**
    - id_holder_permission, id_permission, id_holder, is_permission_active, created_at, updated_at

- **Permission**
    - id_permission, permission_designation, created_at, updated_at

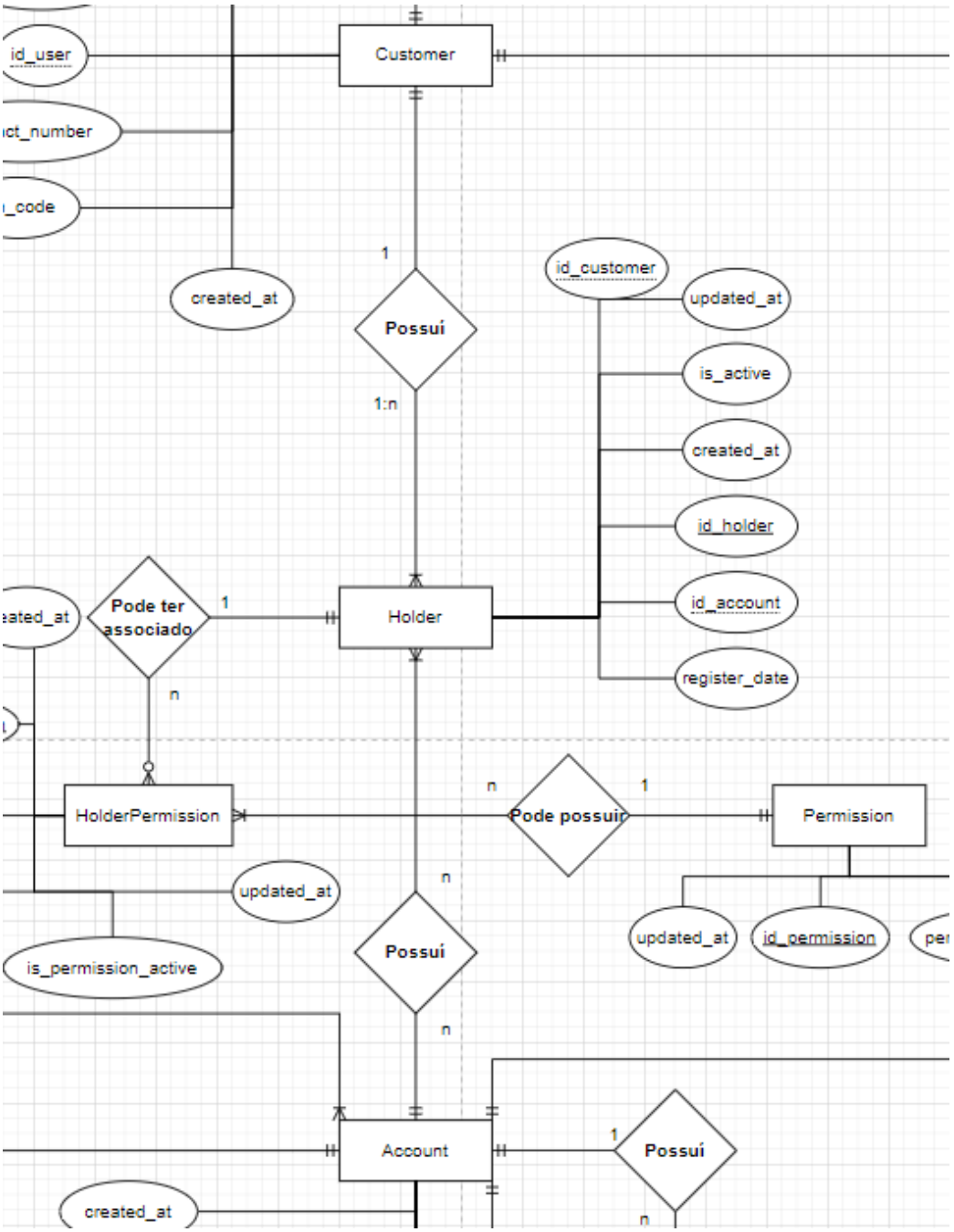


Figura 2: Ligações relativamente ao **Cliente - Conta** e **Titulares - Permissões**

4.1.3 Entidade AccountType associado a Account

Esta associação é relativamente simples. Uma conta possui apenas um tipo de conta associada, enquanto que vários tipos de conta podem estar associados a várias contas.

- De AccountType para Account - **um para muitos**.

Tendo em conta aos seus atributos:

- **Account**
  - id_account, id_account_type, id_credit, balance, is_active ,created_at, updated_at

- **AccountType**
  - id_account_type, account_type_designation, created_at

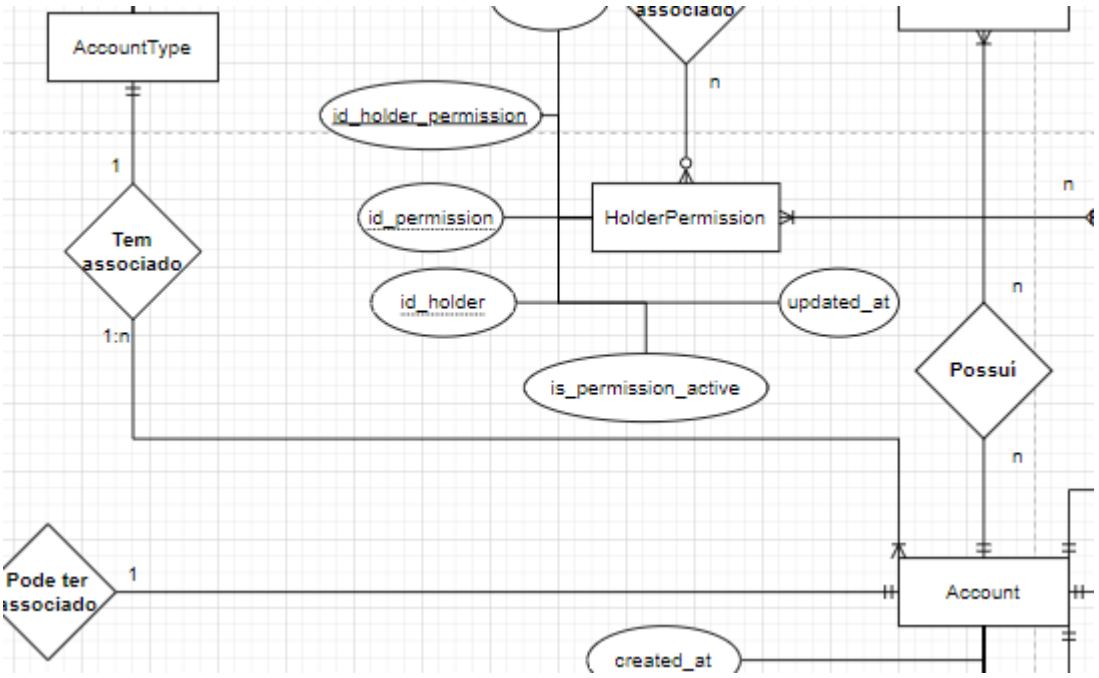


Figura 3: Ligação 1..N entre **AccountType** e **Account**

4.1.4 Entidade Card associado a CardType

Um cartão possui apenas um tipo de tipo de cartão, enquanto que um tipo de cartão pode estar associado a vários cartões.

- **Card**
    - id_card ,id_account, id_card_type, is_active, valid_until, card_pin_number, created_at, updated_at, cvv, card_number, card_holder_name
- **CardType**
    - id_card_type, card_type_designation, created_at

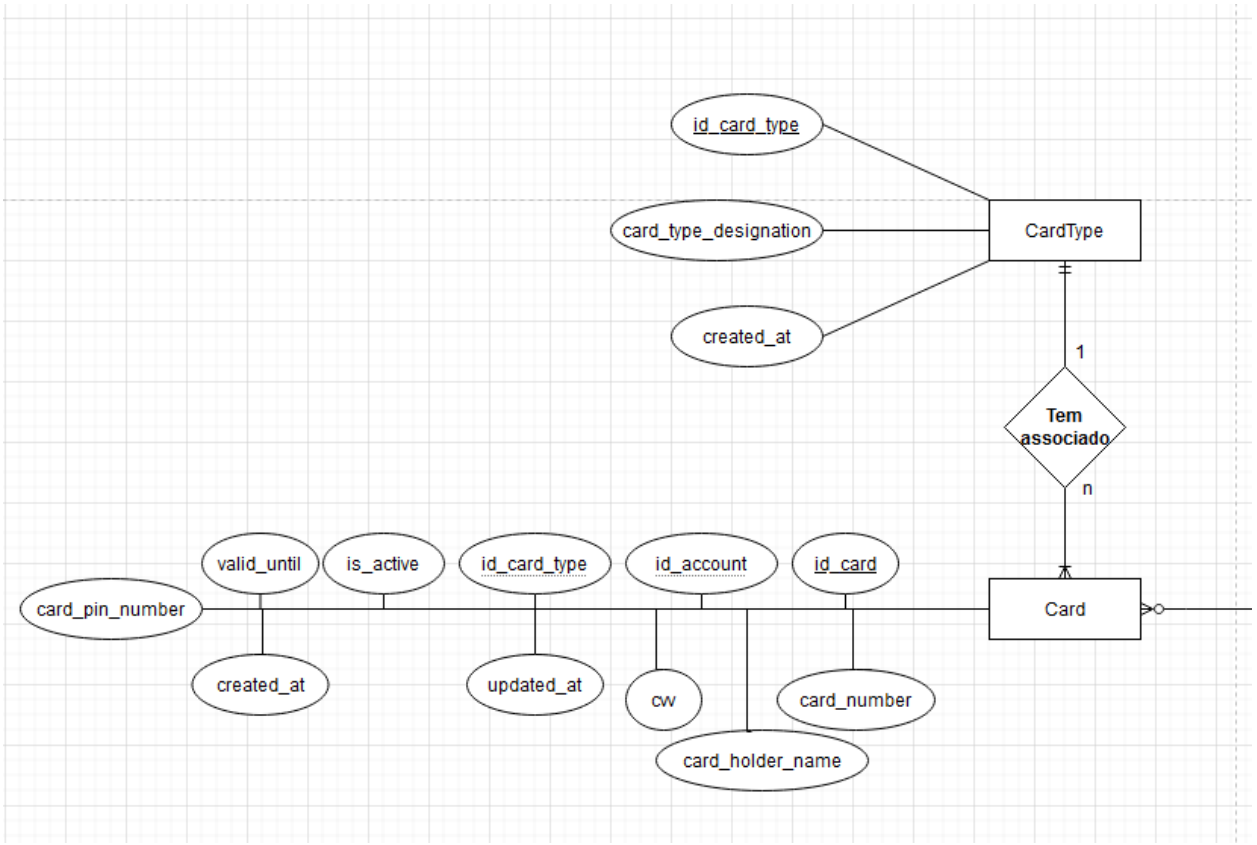


Figura 4: Ligação 1..N entre **CardType** e **Card**

4.1.5 Lógica de Movimentos ligados à Conta

Para idealizar a lógica dos movimentos, foi criada uma ligação muito similar às ligações anteriores.

Uma conta pode possuir vários movimentos associados e um movimento pertence apenas a uma conta. Associado aos movimentos, a entidade em questão possui uma tabela associada que define o tipo de movimento que se pretende.

- **Movement**
    - id_account, reserved_balance, start_balance, id_movement, id_account, id_operaton_type, end_balance, uuid_movement.

- **Operation Type**
    - id_account, reserved_balance, start_balance, id_movement, id_operaton_type, end_balance, uuid_movement, created_at

Para esclarecimento, **o campo reserved_balance**, simboliza o saldo contabilístico, onde é um saldo total que retira todas as nuances de compras pendentes.

O saldo total já será calculado baseado nos pagamentos feitos e com pagamentos autorizados mas que ainda não foram debitados da conta.

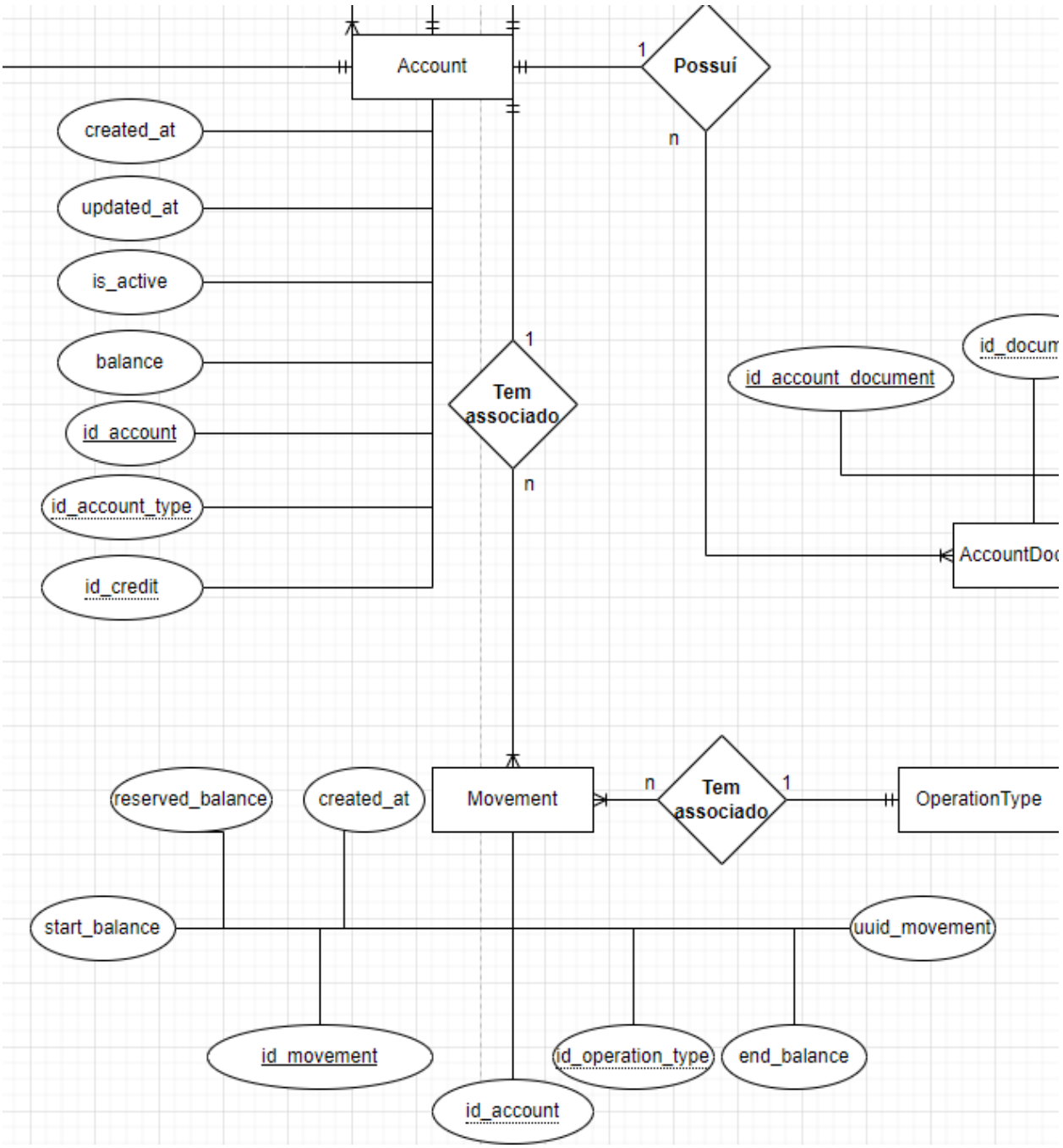


Figura 5: Ligação 1..N entre **Account** e **Movement** e ligação N..1 entre **Movement** e **OperationType**.

4.1.5 Lógica de crédito e pedido de crédito

Para começar a explicar uma das ligações mais fulcrais deste projeto, é necessário referir algumas regras que for

am tido em contas para o desenvolvimento desta lógica:

- Apenas um colaborador será responsável pela avaliação e gestão do mesmo crédito.
- Para simplificar a ideia, é atribuído apenas um status de crédito sem campos exclusivos.

Para se obter um crédito é necessário a criação de um processo inicial, que demanda uma avaliação dos requisitos por parte de um colaborador do banco. Sabendo que um cliente pode pedir 0 ou vários créditos e vários créditos podem ter vários clientes associados, foi criado a tabela **CreditProcess**.

CreditProcess é uma entidade derivada de uma ligação N..N entre a entidade Customer e a entidade Credit.

Com esta ligação, possibilitamos que vários clientes estejam associados ao mesmo crédito, por exemplo, um casal que são do mesmo banco, a pedir um crédito habitação.

Além desta tabela intermediária, foi feita também a associação entre Collaborator e CreditProcess, para que se tenha informação do gestor responsável pelo processo de crédito.

Criado o processo, existe a possibilidade de uma conta ser criada. Normalmente quando se pede um crédito, é associado uma conta ao processo para que seja colocado o montante correspondente ao processo daquela conta.

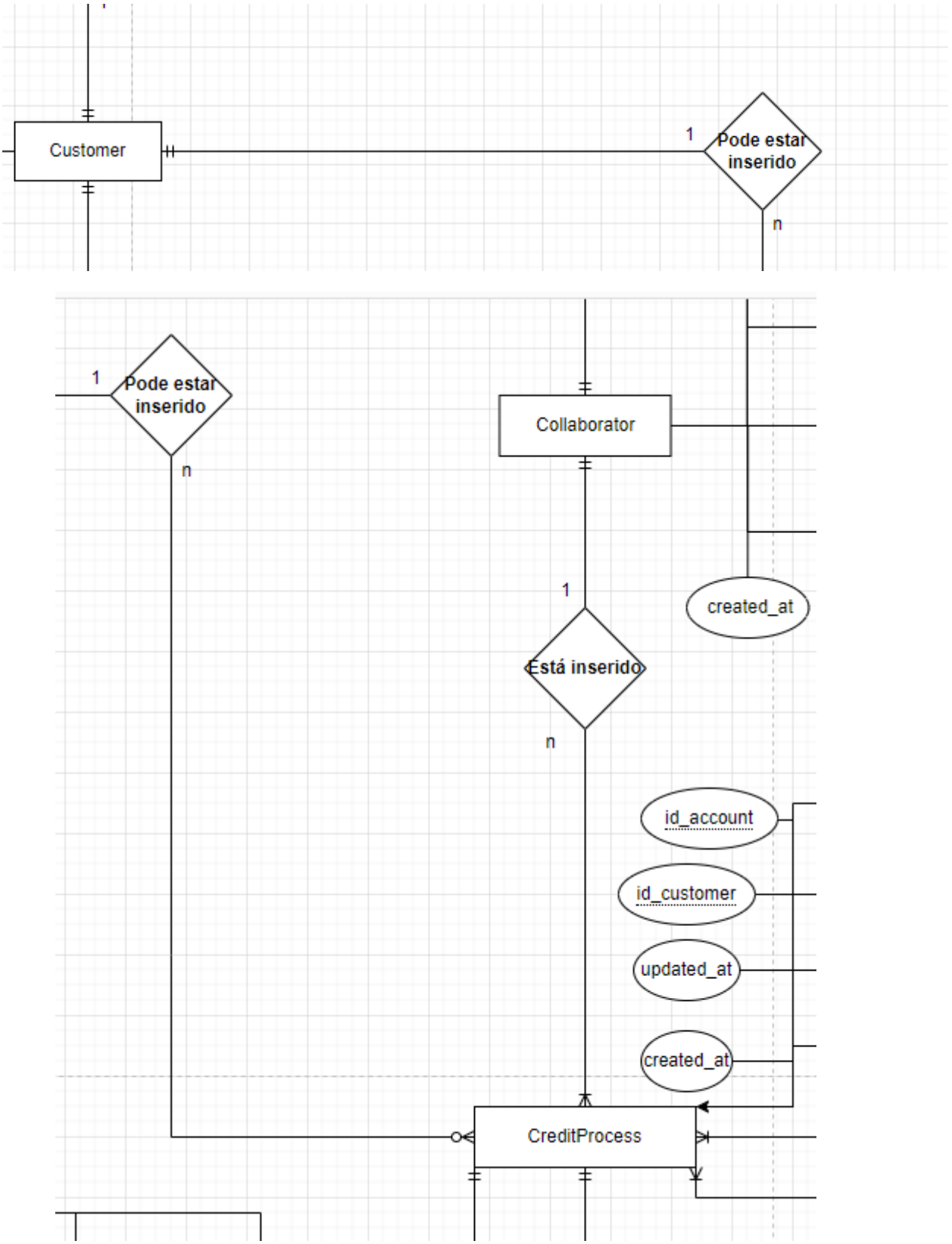
Contudo, ao criar um processo, não necessariamente é obrigatório *a priori* a associação de uma conta ao processo, pois este processo pode ser negado pelo gestor.

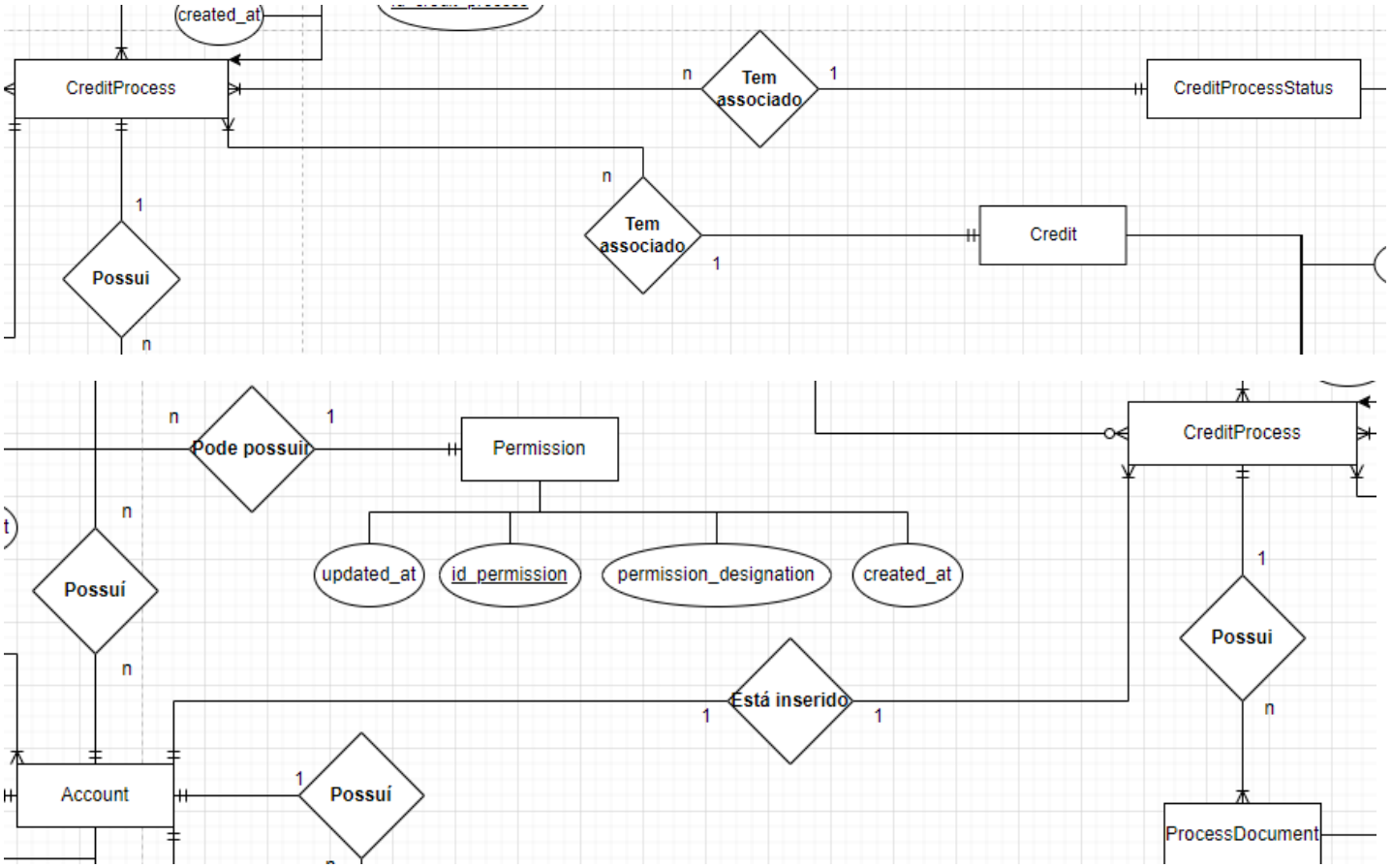
Com base neste pensamento, foi associado à tabela CreditProcess, **uma ligação à entidade Account entre 1..1**, para que se possa ter informação com mais facilidade à conta, além de permitir a criação de uma conta exclusiva para o processo em questão.

- **CreditProcess**
    - id_account, id_credit, id_customer, id_collaborator, updated_at, id_credit_process_status, created_at, id_credit_process

- **Credit**
    - id_credit, time_remaining, amount, created_at, is_paid, initial_duration, updated_at

- **CreditProcessStatus**
    - id_credit_process_status, status, created_at


Customer
1
Pode estar inserido
n
1
Pode estar inserido
n
Collaborator
1
Está inserido
n
created_at
id_account
id_customer
updated_at
created_at
CreditProcess



Figuras 6, 7 e 8 e 9: Ligações **1..N** entre Customer e CreditProcess**, N..1** entre CreditProcess e Credit e **1..N** entre Account e CreditProcess.

4.1.6 Lógica de documentos associados a Contas e Processos

Complementando a ideia dos processos e das contas, foi pensado acrescentar uma funcionalidade de documentos.

Uma conta poderá ter documentos exclusivos porque o cliente não necessariamente precisa de um processo de crédito para ter uma conta. Documentos pessoais, vencimentos e/ou recibos, autoridade tributária, etc. Algo que seja importante para a criação de uma conta.

Assim como a conta,pode ter documentos exclusivos quando se pede um crédito obrigatório. Normalmente, nestes casos em concreto, podem ser pedidos documentos exclusivos que possam ser de grande importância para o processo, como por exemplo: fotos de um imóvel, apresentação da caderneta predial, documentos da situação tributária, entre outros documentos.

Para isso, foi criado uma tabela Document com ligações **N..N** para Account e ProcessDocument.

O motivo para existir estas tabelas intermediárias serve para organizar os documentos por exclusividade, ou seja, separar quais documentos são e onde é que estão associadas, evitando assim, duplicar informação dos documentos, caso este esteja presente tanto numa conta quando num processo de crédito.

- **Document**
    - id_document, url, created_at, document_name, updated_at

- **ProcessDocument**
    - id_process_document, id_document, id_credit_process

- **AccountDocument**
    - id_account_document, id_document, id_account

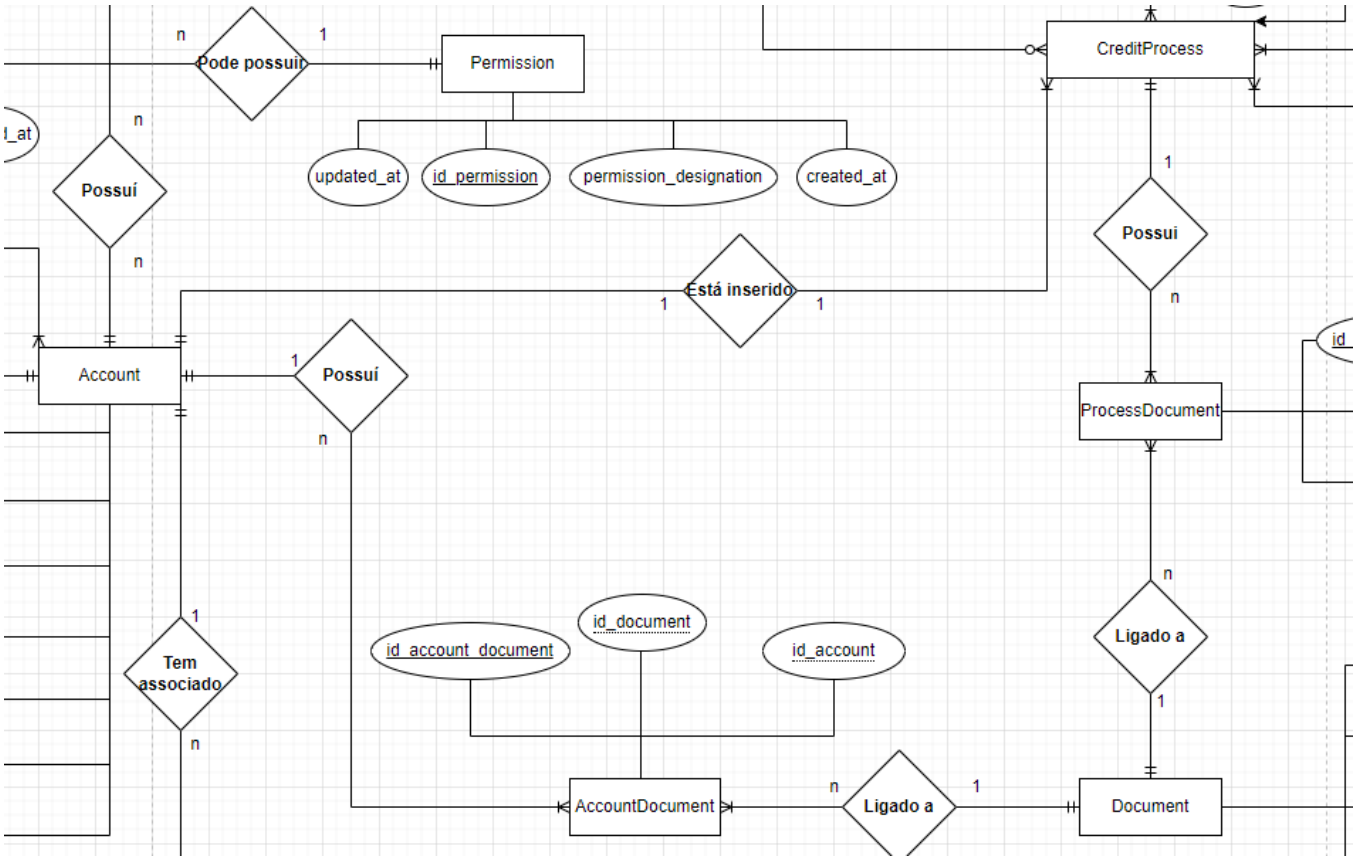


Figura 10: Ligações **1..N** entre Account e AccountDocument**, N..1** entre AccountDocument e Document, **1..N** entre CreditProcess e ProcessDocument e **N..1** entre ProcessDocument e Document.

4.2 Modelo Relacional

O modelo relacional que foi concretizado a partir do Modelo Entidade Associação - EA.

Existem tabelas cujas as suas chaves são primárias **(Primary Keys)** e estrangeiras **(Foreign Key)** ao mesmo tempo. A este acontecimento, estas chaves são designadas como **Chave Compostas**. Para este exercício/projeto, estão configuradas para chaves estrangeiras, ao nível do código, contudo, será tratado teoricamente como chaves compostas

Como já referido anteriormente sobre as relações entre chaves e os seus atributos, abaixo, está presente uma tabela com um resumo de todas as ligações: primárias, estrangeiras e compostas.

Estas chaves compostas foram criadas em todas as tabelas de ligações **N..N** de modo a evitar que não seja possível a repetição de um certo par de chaves, conforme as regras de normalização de tabelas.

Tables	Primary key's	Foreign Key's	Composite Key's
User	id_user	-	-
Customer	id_customer	id_user	-
Collaborator	id_collaborator	id_user	-
CreditProcess	id_credit_process	id_collaborator id_credit_process_status id_account	id_customer id_credit
CreditProcessStatus	id_credit_process_status	-	-
Credit	id_credit	-	-
Holder	id_holder	-	id_customer id_account
HolderPermission	id_holder_permission id_permission id_holder	id_permission id_holder	id_permission id_holder
Permission	id_permission	-	
Account	id_account	id_account_type	-
AccountType	id_account_type	-	-
ProcessDocument	id_process_document	-	id_document id_credit_process
Document	id_document	-	-
AccountDocument	id_account_document	-	id_account id_document
Card	id_card	id_card_type	-
CardType	id_card_type	-	-
Movement	id_movement	id_account id_operation_type	-
OperationType	id_operation_type	-	-

Figura 11: Tabela informativa com todas as chaves primárias, estrangeiras e compostas.

6. Conclusão

Com o esforço conjunto conseguiu-se aprender, não só, uns com os outros, como funciona melhor a idealização e construção de uma base de dados mas também, o aperfeiçoamento da habilidade de se conseguir dividir um problema em pequenos pedaços para no final chegar ao objetivo final.

A divisão entre o Modelo Entidade Associação e o modelo Relacional foi originada a partir desse pressuposto.

O grupo sentiu, contudo, algumas dificuldades ao longo do processo, à medida que as tabelas iam crescendo pelo facto de não haver, por vezes, alguma calma e tranquilidade na hora de pensar no problema. Com "cabeça e paciência", o resultado poderia ter sido mais lapidado e simples, acreditando sempre em uma possível melhoria da solução atual, tornando-se assim, a sugestão principal de melhoria do presente projeto.

5. Anexos

1. **Modelos Relacional e EA**

   https://viewer.diagrams.net/?tags=%7B%7D&highlight=0000ff&edit=_blank&layers=1&nav=1&title=Untitled%20Diagram.drawio#Uhttps%3A%2F%2Fraw.githubusercontent.com%2Fpedro7161%2FCiencia_dos_Dados_TP1_Base-de-dados%2Fmaster%2FUntitled%2520Diagram.drawio