

Bases de Dados

PL06 – Modelação Lógica e
Normalização

Docente: Diana Ferreira

Email: diana.ferreira@algoritmi.uminho.pt

Horário de Atendimento:

5ª feira 16h–17h



Sumário

1

Regras de Derivação

2

Modelo Relacional

3

Normalização

4

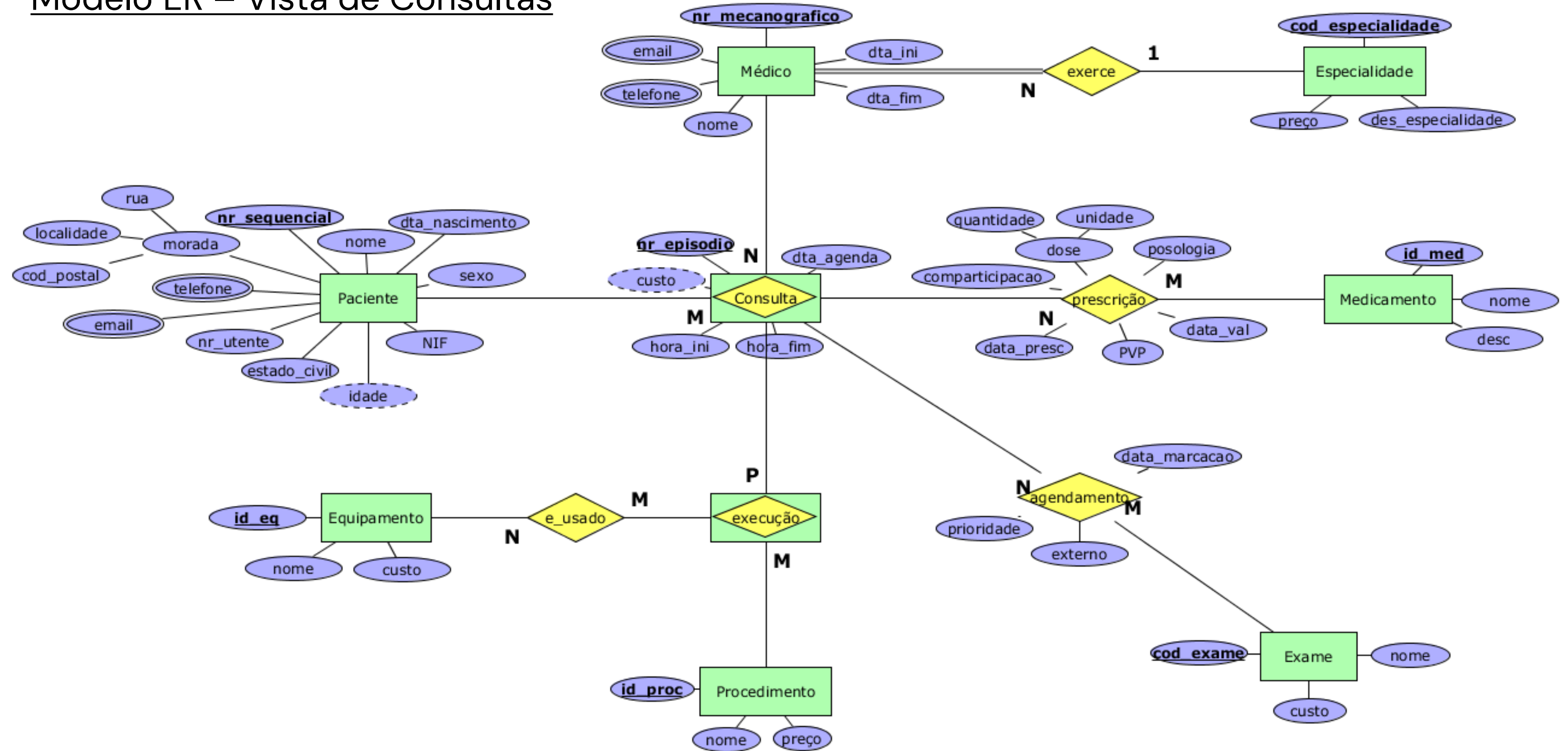
Álgebra Relacional

Bibliografia:

- Connolly, T., Begg, C., Database Systems, A Practical Approach to Design, Implementation, and Management , Addison-Wesley, 4a Edição, 2004. **(Chapter 17; Chapter 4/5; Chapter 14/15)**
- Teorey, T., Database Modeling and Design: The Fundamental Principles, II Edição, Morgan Kaufmann, 1994.
- Belo, O., “Bases de Dados Relacionais: Implementação com MySQL”, FCA – Editora de Informática, 376p, Set 2021. ISBN: 978-972-722-921-5.

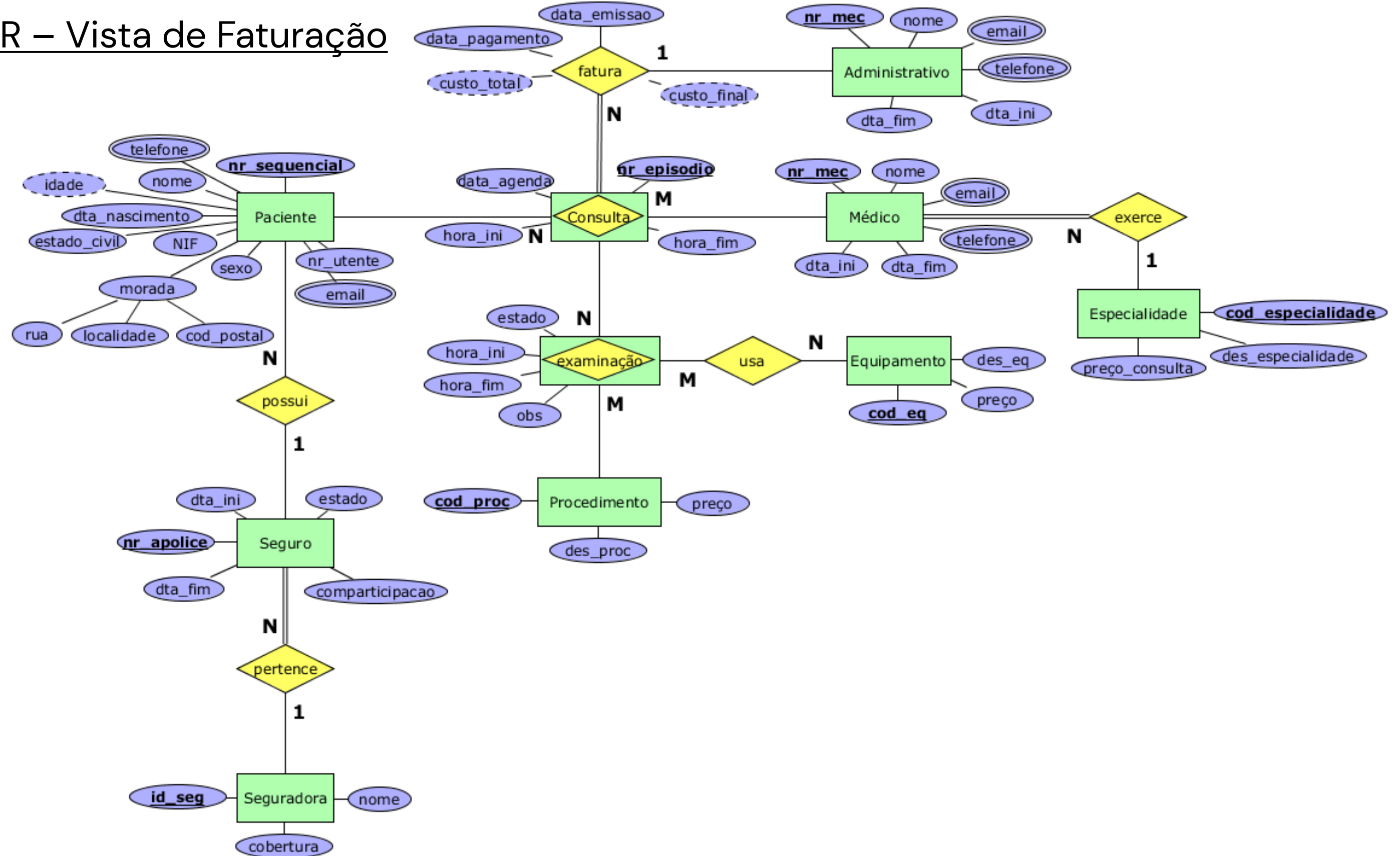
Revisão da aula anterior

Modelo ER – Vista de Consultas



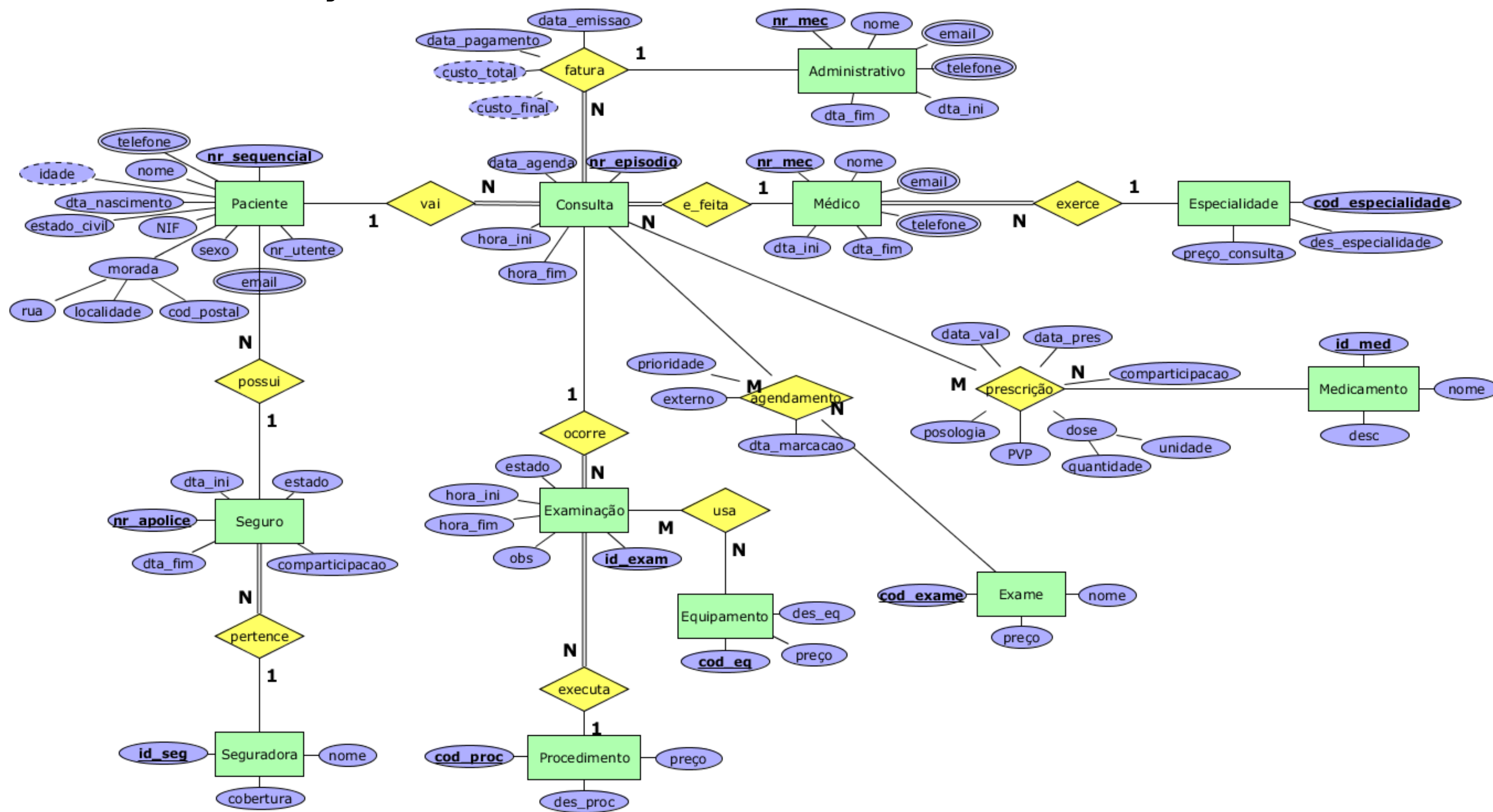
Revisão da aula anterior

Modelo ER – Vista de Faturação



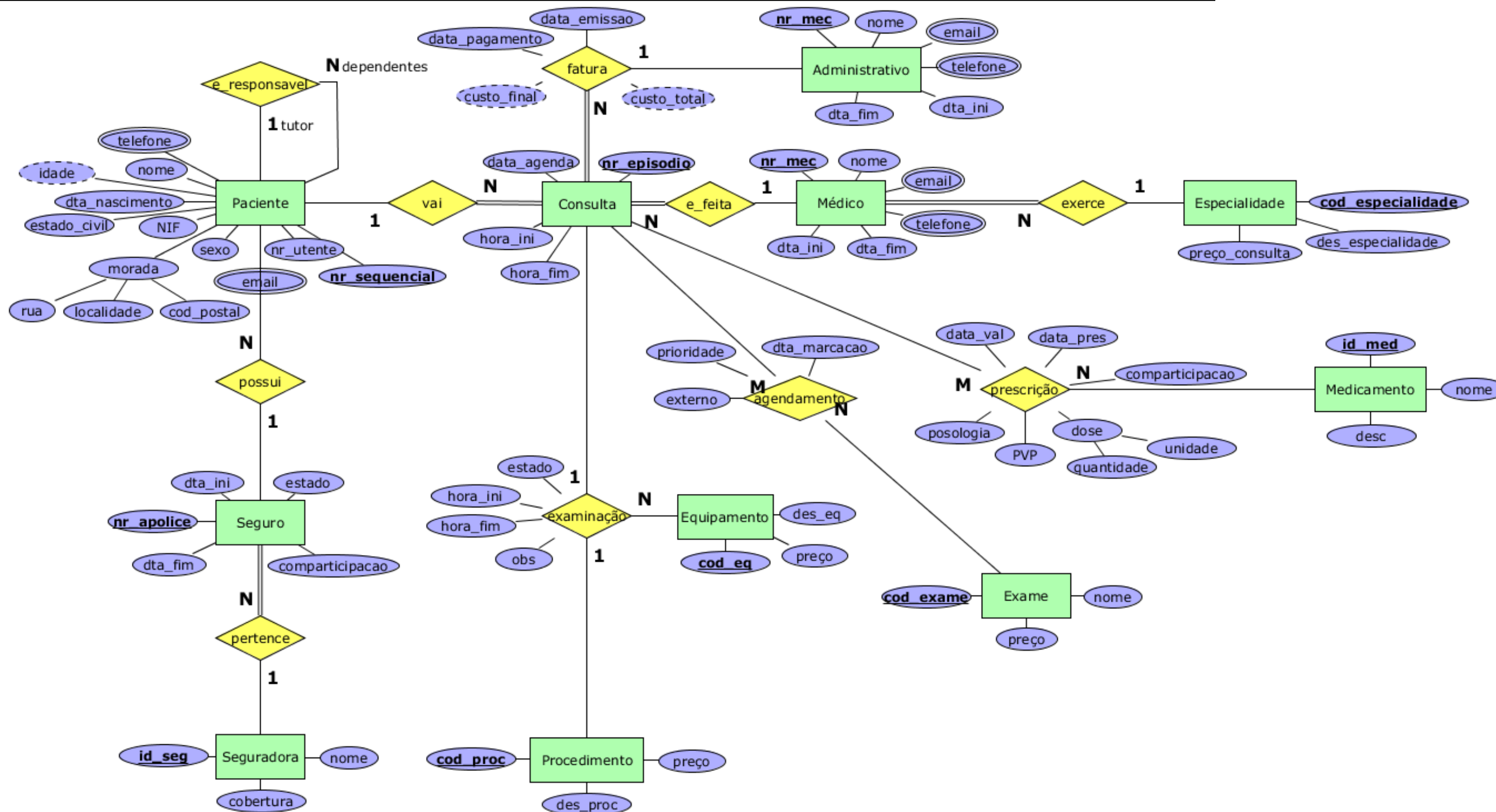
Revisão da aula anterior

Modelo ER – Combinação das Vistas



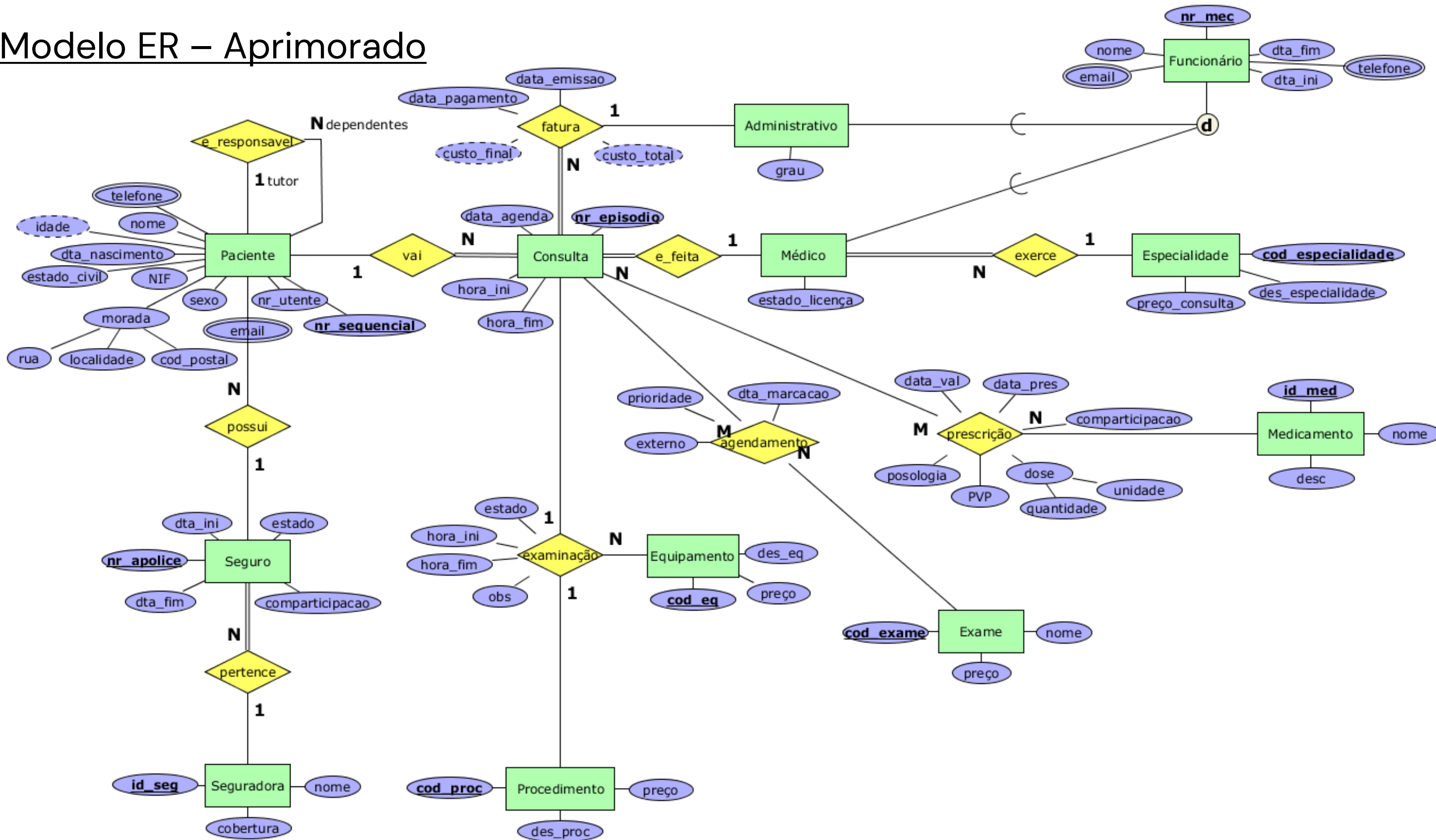
Modelo ER – Com relacionamento recursivo + relacionamento ternário

Modelo ER – Com relacionamento recursivo + relacionamento ternário



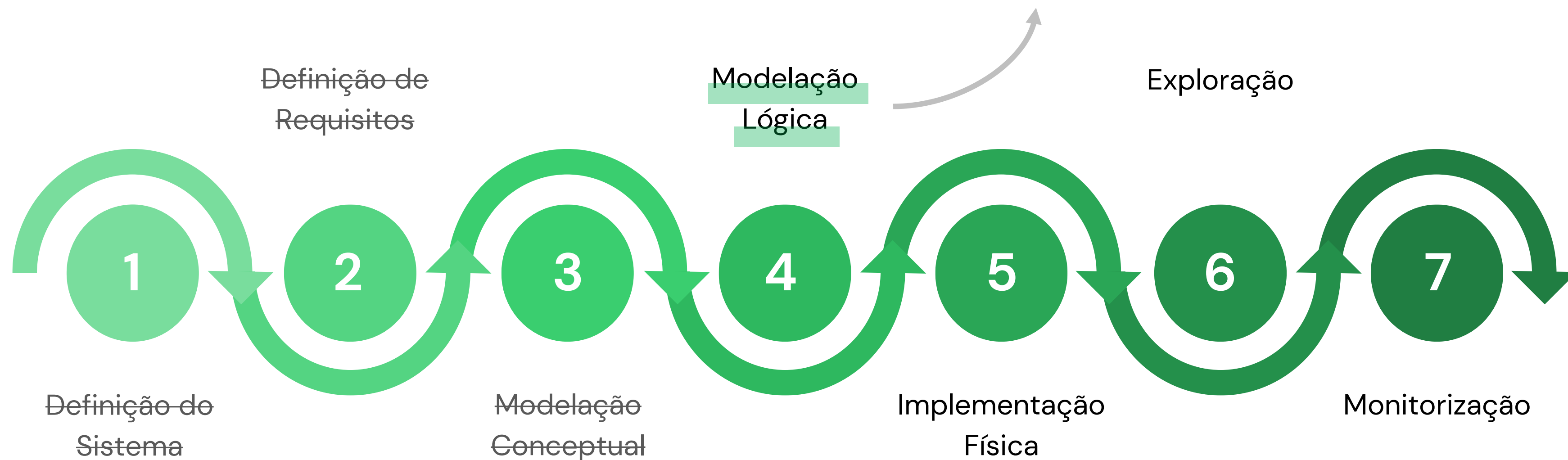
Revisão da aula anterior

Modelo ER – Aprimorado

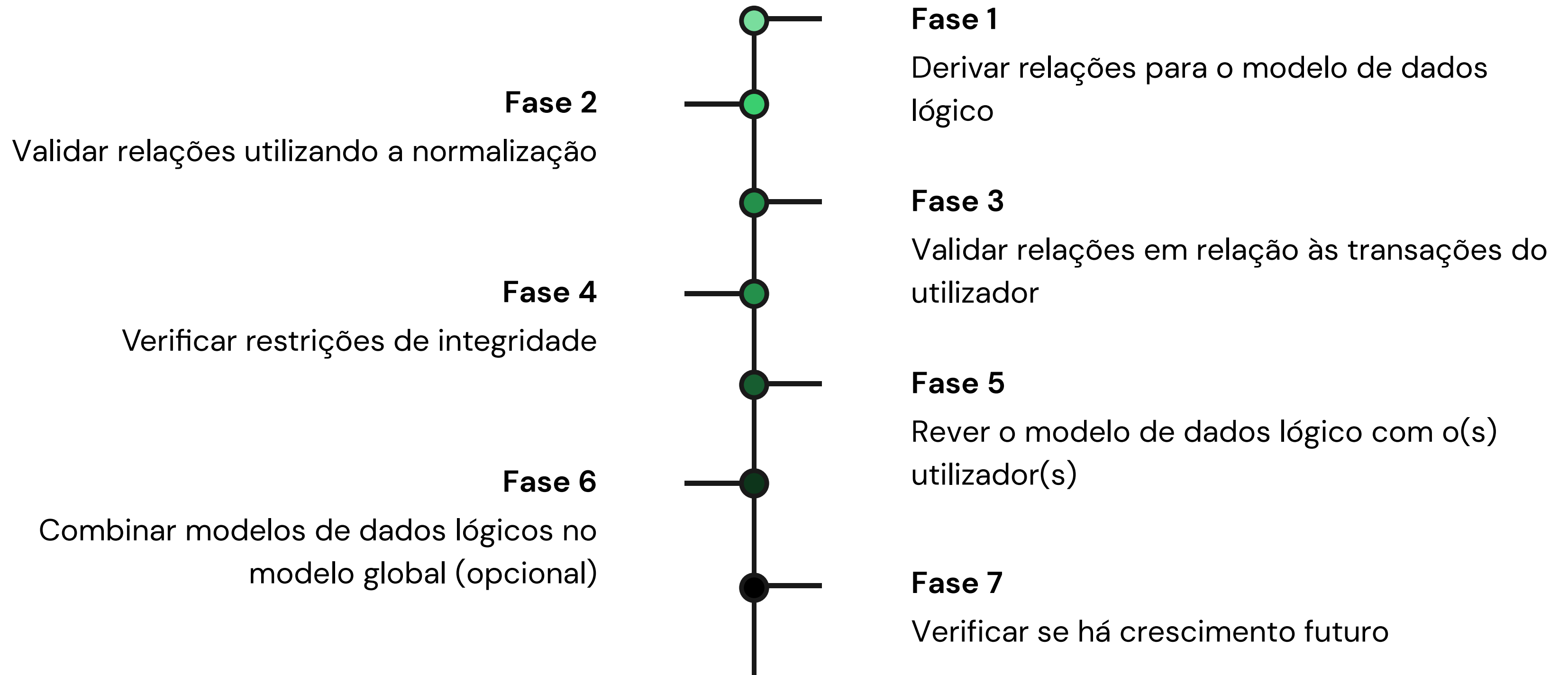


Ciclo de vida de um SBD

Traduzir o modelo de dados conceptual num modelo de dados lógico e, em seguida, validar o modelo para verificar se este é estruturalmente correto e capaz de suportar as transações necessárias.



Ciclo de vida de um SBD: Modelação Lógica



Modelo Relacional

Modelo lógico para BDs relacionais, baseado no conceito de relação, também designado por tabela.

Modelação Física

O modelo relacional pode depois ser concretizado num SGBD usando a linguagem SQL.

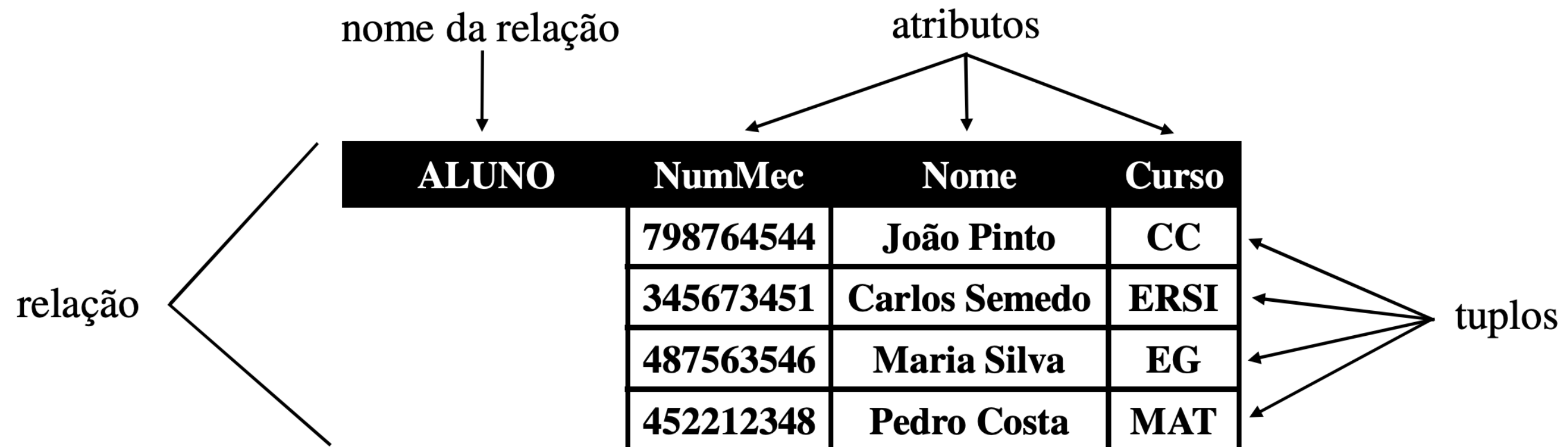
Modelação Lógica

As entidades-tipo e relacionamentos do modelo ER são mapeados em relações/tabelas no modelo relacional.

Modelação Conceptual

Modelo Relacional

- É baseado no conceito de **relação**, onde uma relação é uma **tabela** de valores.
- Uma tabela de valores pode ser vista como um conjunto de **linhas, registos** ou **tuplos**.
- Cada tuplo é identificado por um conjunto de **colunas, campos** ou **atributos**.
- Uma base de dados é representada como um conjunto de relações.



FASE 4: Modelação Lógica

→ Derivar relações

- O relacionamento que uma entidade tem com outra entidade é representado pelo mecanismo de **chave primária/chave estrangeira**.
- Para decidir onde colocar o(s) atributo(s) de chave estrangeira, devemos primeiro identificar as entidades '**pai**' e '**filho**' envolvidas no relacionamento.
- A entidade **pai** refere-se à entidade que **envia uma cópia da sua chave primária** na relação que representa a entidade **filho**, para atuar como a **chave estrangeira**.

FASE 4: Modelação Lógica

➔ Derivar relações

O processo de derivação passa por descrever como as relações são derivadas para as seguintes estruturas que podem ocorrer num modelo de dados concetual:

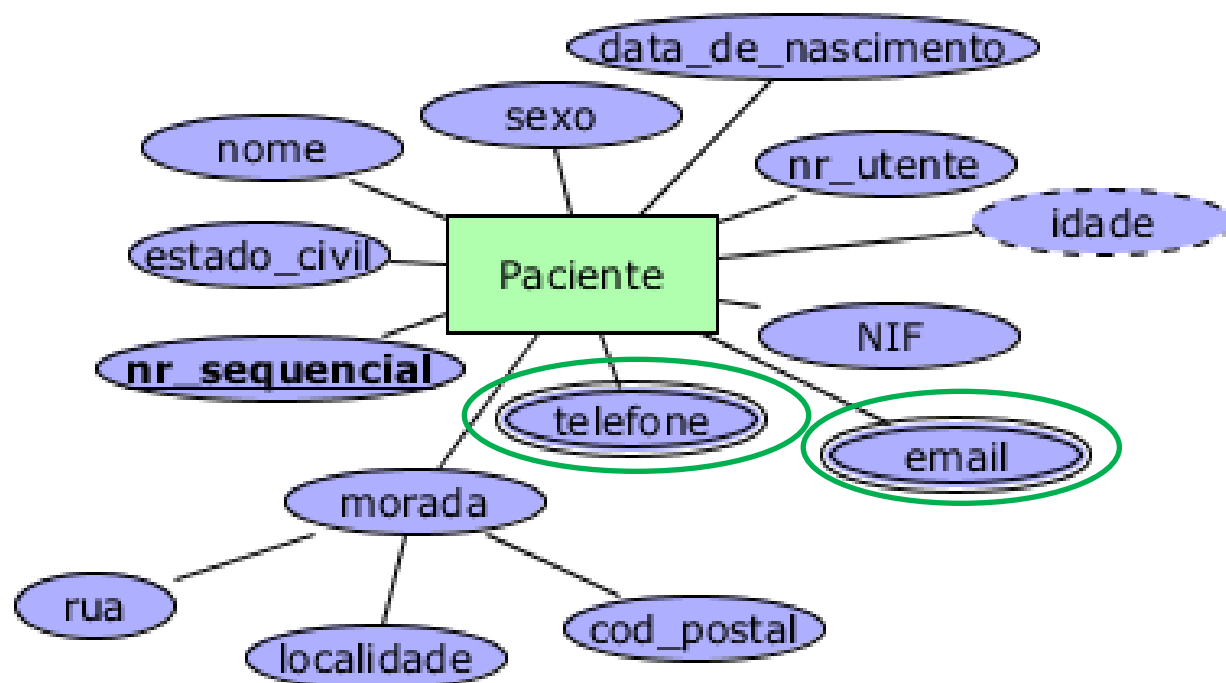
- Entidades Simples
- Atributos multivalor
- Entidades Fracas
- Relacionamentos binários de um-para-muitos (1:N)
- Relacionamentos binários de muitos-para-muitos (N:M)
- Entidade Relacionamento
- Relacionamentos binários de um-para-um (1:1)
- Relacionamentos binários recursivos de um-para-um (1:1)
- Relacionamentos complexos
- Relacionamentos superclasse/subclasse

FASE 4: Modelação Lógica

→ Derivar relações

▪ Entidades Simples

Para cada entidade do modelo de dados, crie **uma relação/tabela** que inclua todos os **atributos simples** dessa entidade. Os atributos derivados devem ser analisados e no caso dos atributos compostos, são apenas incluídos os atributos simples constituintes.



Paciente (nr_sequencial, nome, sexo, dta_nascimento, rua, localidade, cod_postal, NIF, nr_utente, estado_civil)

Chave primária nr_sequencial

Chave candidata NIF

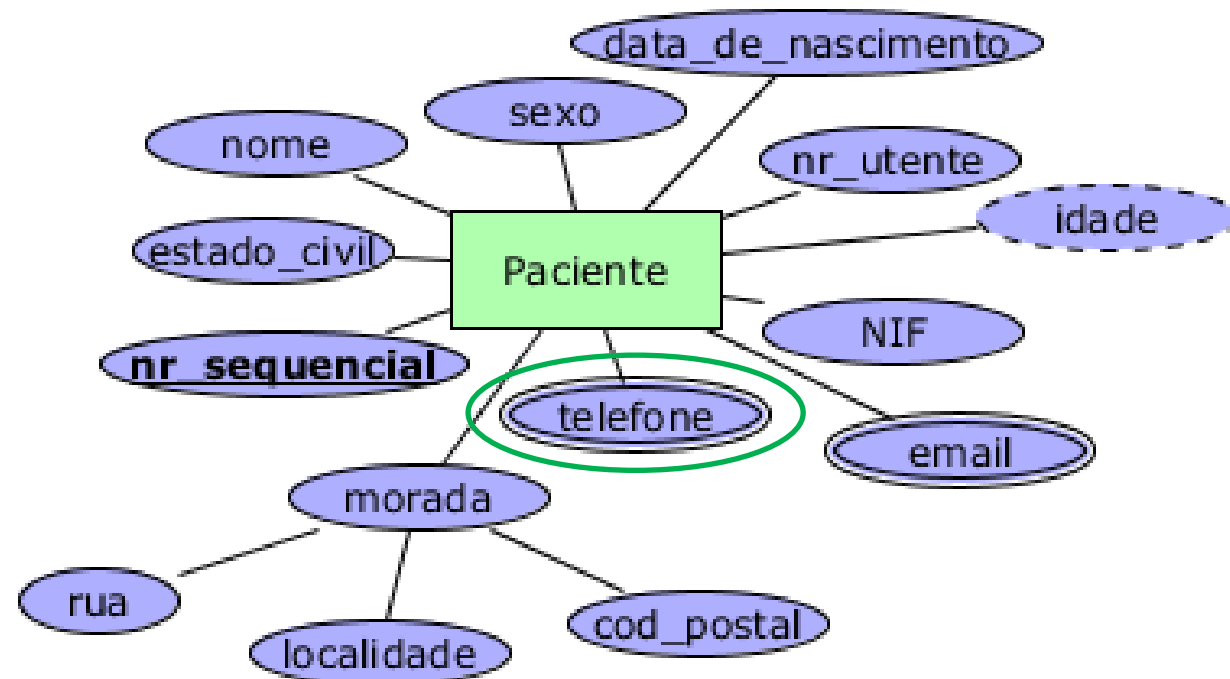
Chave candidata nr_utente

Derivado idade(dta_atual – dta_nascimento)

Paciente
<u>nr_sequencial</u>
nome
sexo
dta_nascimento
rua
localidade
cod_postal
NIF
nr_utente
estado_civil

➡ Derivar relações

Para cada atributo **multivalor**, crie uma **nova relação** para representar o atributo **multi-valor** com relacionamento de **1:N** com a sua tabela de referência e inclua a **chave primária** da entidade na nova relação, para atuar como **chave estrangeira**.



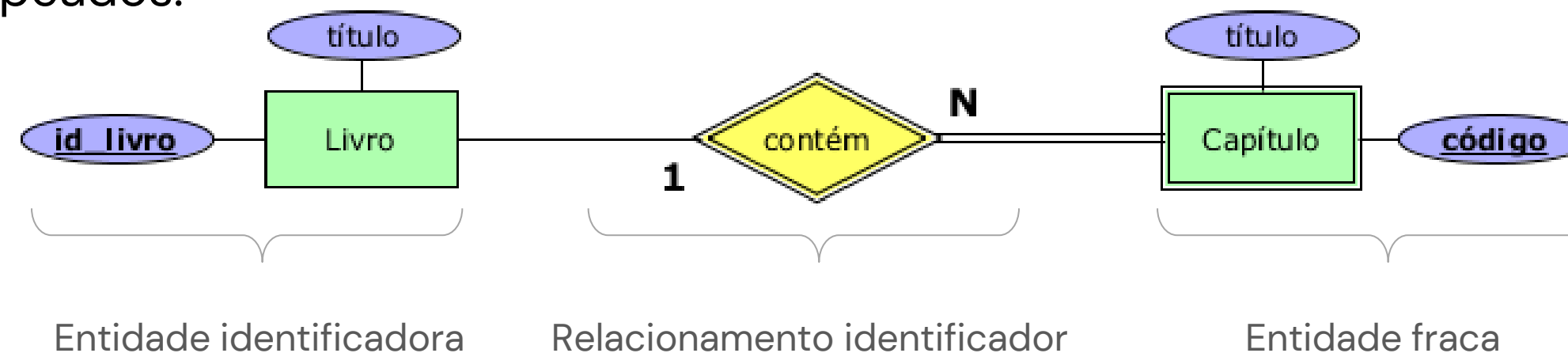
Paciente(nr_sequencial)

FASE 4: Modelação Lógica

→ Derivar relações

▪ Entidades Fracas

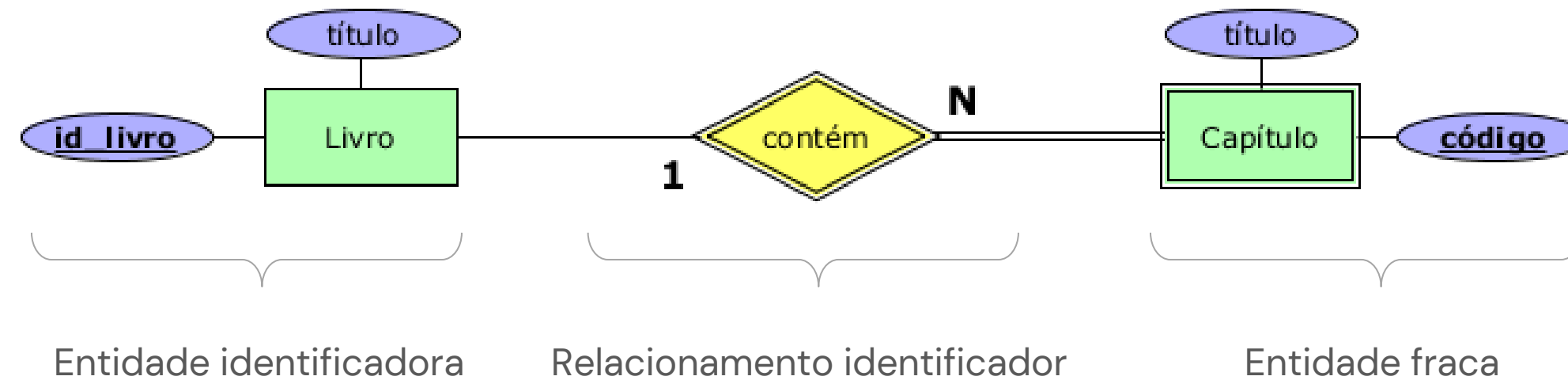
- Para cada entidade fraca do modelo de dados, crie uma relação que inclua todos os atributos simples dessa entidade.
- Se a entidade fraca não possuir atributos que possam constituir chaves candidatas, o conjunto de atributos que permitem identificar univocamente uma ocorrência da entidade fraca, é a **chave parcial** da entidade fraca;
- A chave primária de uma entidade fraca é sempre uma **chave composta** da chave primária da entidade identificadora e da sua chave parcial, portanto, a identificação da chave primária de uma entidade fraca não pode ser feita até que todos os relacionamentos com as entidades proprietárias tenham sido mapeados.



FASE 4: Modelação Lógica

→ Derivar relações

▪ Entidades Fracas



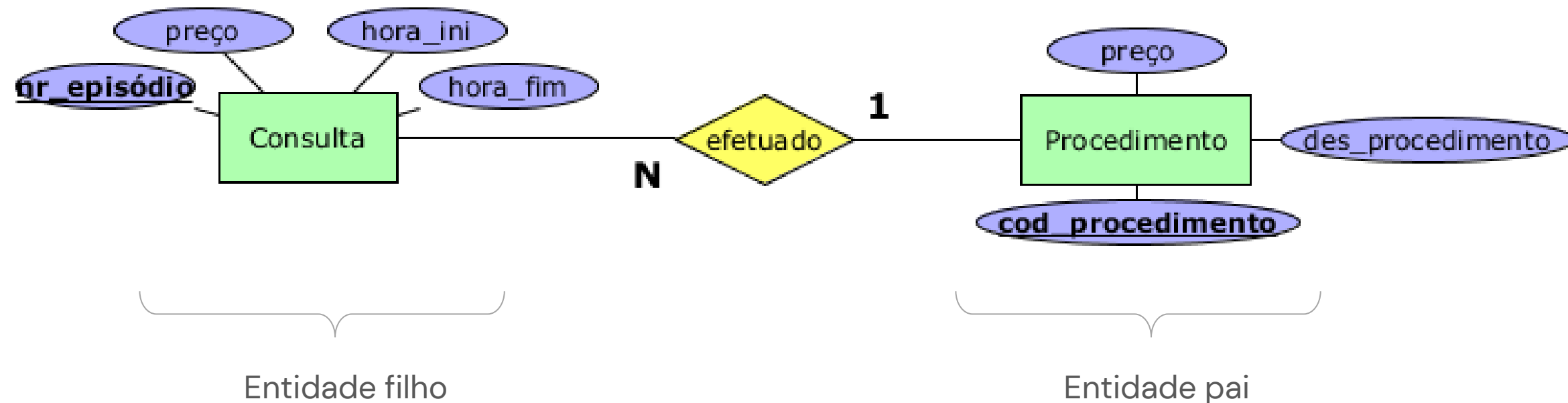
Livro (id_livro, título)
Chave primária id_livro

Capítulo (id_livro, codigo, título)
Chave primária id_livro, codigo
Chave estrangeira id_livro **referencia** Livro(id_livro)

FASE 4: Modelação Lógica

➔ Derivar relações

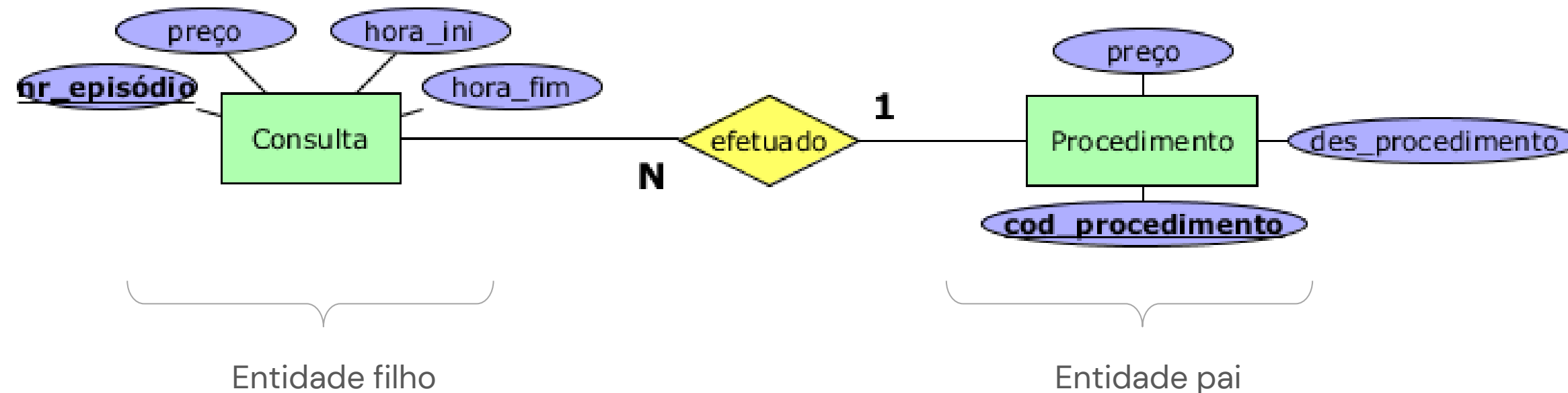
- Relacionamentos binários de um-para-muitos (1:N)
 - Para cada relacionamento binário 1:N, a entidade do lado 'um' do relacionamento é designada como a entidade pai e a entidade do lado 'muitos' é designada como a entidade filho.
 - Para representar esse relacionamento, cria-se uma **cópia** do(s) atributo(s) de **chave primária** da **entidade pai** na relação que representa a **entidade filho**, para atuar como **chave estrangeira**.



FASE 4: Modelação Lógica

→ Derivar relações

- Relacionamentos binários de um-para-muitos (1:N)



Consulta (nr_episodio, preço, hora_ini, hora_fim, cod_procedimento)

Chave primária nr_episodio

Chave Estrangeira cod_procedimento referencia

Procedimento(cod_procedimento)

Procedimento (cod_procedimento, des_procedimento, preço)

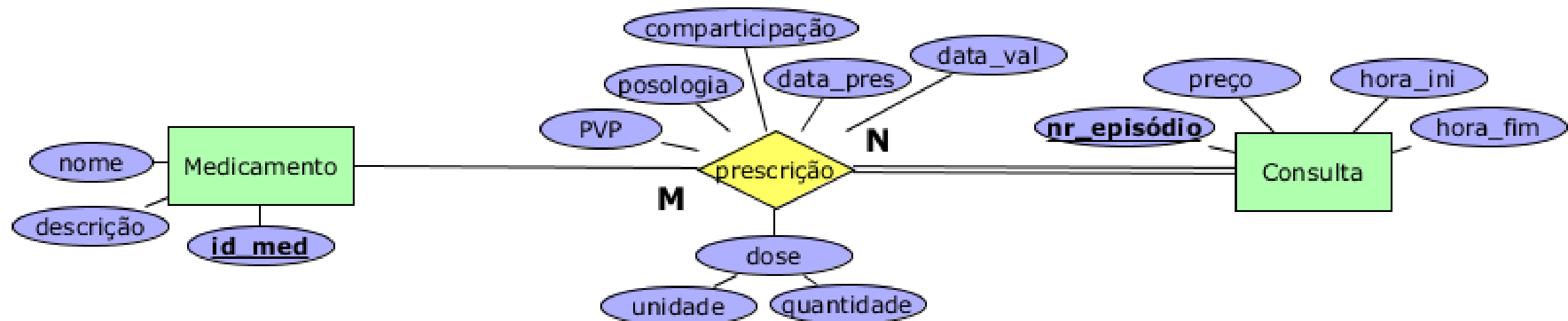
Chave primária cod_procedimento

Chave primária cod_procedimento

FASE 4: Modelação Lógica

→ Derivar relações

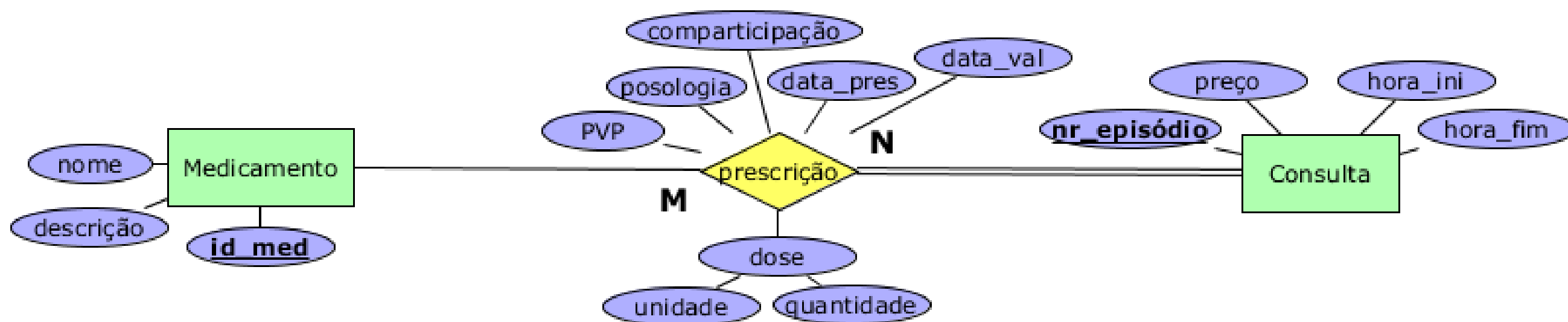
- Relacionamentos binários de muitos-para-muitos (N:M)
 - Crie uma relação para representar o relacionamento e inclua quaisquer atributos que façam parte do relacionamento.
 - Crie uma **cópia** do(s) atributo(s) de **chave primária** das **entidades** que participam no relacionamento na nova relação, para atuar como **chaves estrangeiras**. A **chave primária** da nova relação é sempre uma chave composta pelas chaves estrangeiras, possivelmente em combinação com outros atributos do relacionamento.



FASE 4: Modelação Lógica

→ Derivar relações

- Relacionamentos binários de muitos-para-muitos (N:M)



Medicamento (id_med, nome, descrição)
Chave primária id_med

Consulta (nr_episodio, preço, hora_ini, hora_fim)
Chave primária nr_episodio

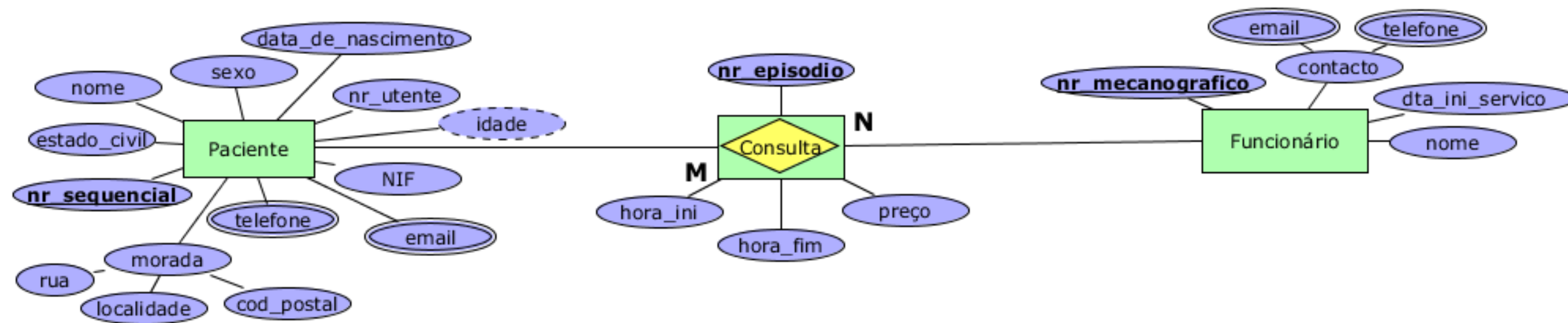
Prescrição (id_med, nr_episodio, unidade, quantidade, posologia, PVP, participação, data_val, data_pres)
Chave primária id_med, nr_episodio
Chave Estrangeira id_med **referencia** Medicamento(id_med)
Chave Estrangeira nr_episodio **referencia** Consulta(nr_episodio)

FASE 4: Modelação Lógica

➔ Derivar relações

▪ Entidade Relacionamento

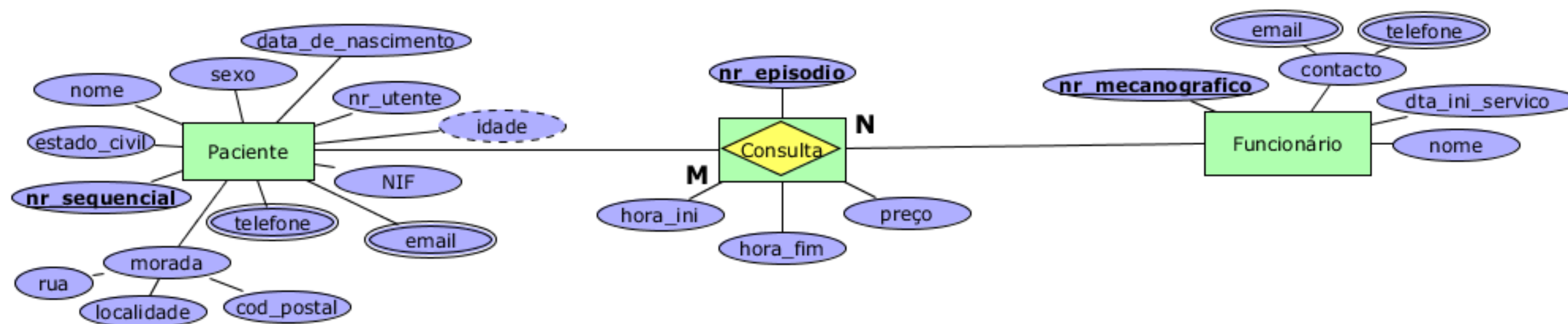
- Crie uma relação para representar a entidade-relacionamento como se fosse uma entidade independente e inclua todos os atributos que fazem parte da entidade-relacionamento.
- Crie uma **cópia** do(s) atributo(s) de **chave primária** das **entidades** que participam na entidade-relacionamento na nova relação, para atuar como **chaves estrangeiras**. Caso a entidade-relacionamento **não** possua chave primária, essas chaves estrangeiras formarão a **chave primária**.



FASE 4: Modelação Lógica

→ Derivar relações

▪ Entidade Relacionamento



Paciente (nr_sequencial, nome, sexo, dta_nascimento, rua, localidade, cod_postal, NIF, nr_utente, estado_civil)

Chave primária nr_sequencial

Chave candidata NIF

Chave candidata nr_utente

Derivado idade(dta_atual – dta_nascimento)

Consulta (nr_episodio, nr_sequencial, nr_mecanografico, hora_ini, hora_fim, preco)

Chave primária nr_episodio

Chave Estrangeira nr_sequencial **referencia** Paciente(nr_sequencial)

Chave Estrangeira nr_mecanografico **referencia** Funcionário(nr_mecanografico)

Funcionário (nr_mecanografico, nome, dta_ini_servico)

Chave primária nr_mecanografico

FASE 4: Modelação Lógica

→ Derivar relações

- Relacionamentos binários de um-para-um (1:1)
 - Nestes casos, a criação de relações é mais complexa, porque a **cardinalidade** não pode ser usada para identificar as entidades pai e filho num relacionamento.
 - Em vez disso, as restrições de **participação** são usadas para decidir se é preferível combinar as entidades numa só relação ou se é mais adequado criar duas relações e colocar uma cópia da chave primária de uma relação na outra:
 - (a) participação obrigatória em ambos os lados do relacionamento 1:1;
 - (b) participação obrigatória num lado do relacionamento 1:1;
 - (c) participação opcional em ambos os lados do relacionamento 1:1.

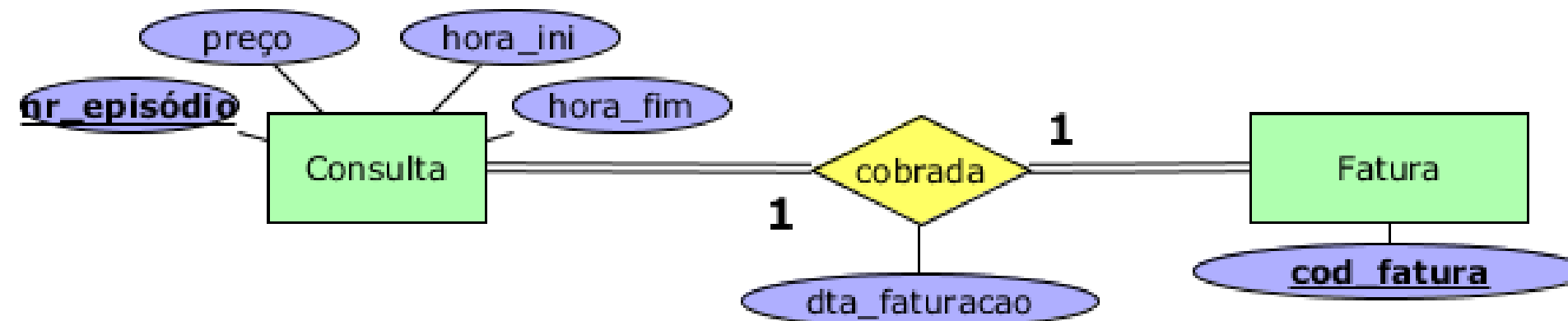
FASE 4: Modelação Lógica

→ Derivar relações

▪ Relacionamentos binários de um-para-um (1:1)

→ (a) participação obrigatória em ambos os lados do relacionamento 1:1;

– Combinar as entidades envolvidas **numa só relação** e escolher uma das chaves primárias das entidades originais para ser a chave primária da nova relação, enquanto outra (se existir) é usada como chave candidata.



Consulta (nr_episodio, preço, hora_ini, hora_fim, dta_faturacao, cod_fatura)

Chave primária nr_episodio

Chave candidata cod_fatura

FASE 4: Modelação Lógica

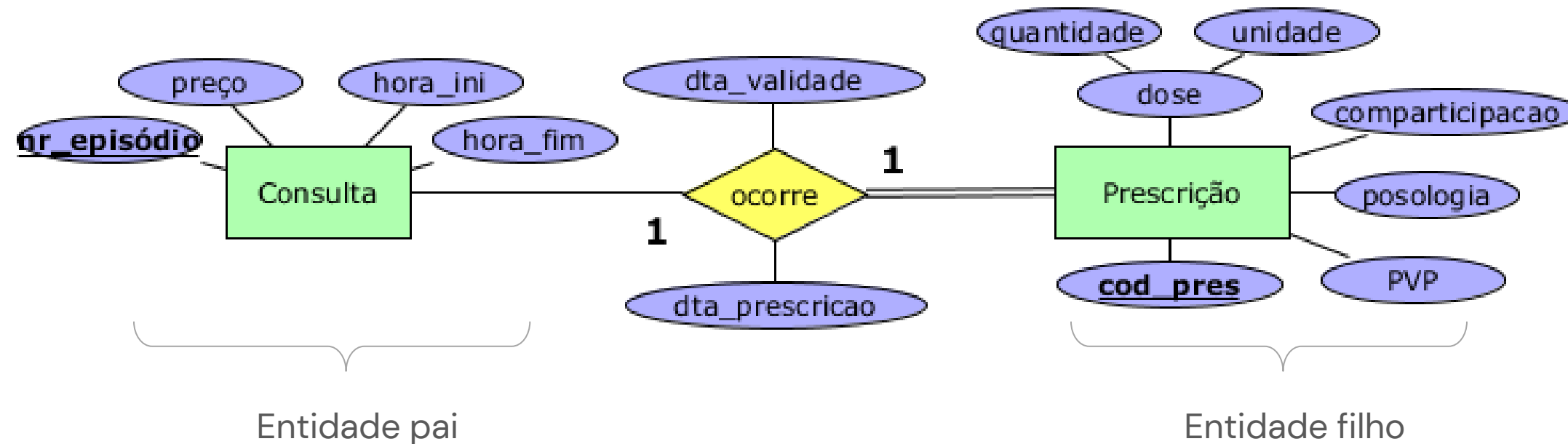
→ Derivar relações

- Relacionamentos binários de um-para-um (1:1)

→ (b) participação obrigatória num lado do relacionamento 1:1;

- A entidade com **participação opcional** é designada como **entidade-pai** e a entidade com **participação obrigatória** como **entidade-filho**.

- **Cópia** da **chave primária** da **entidade pai** colocada na relação que representa a **entidade filho**.



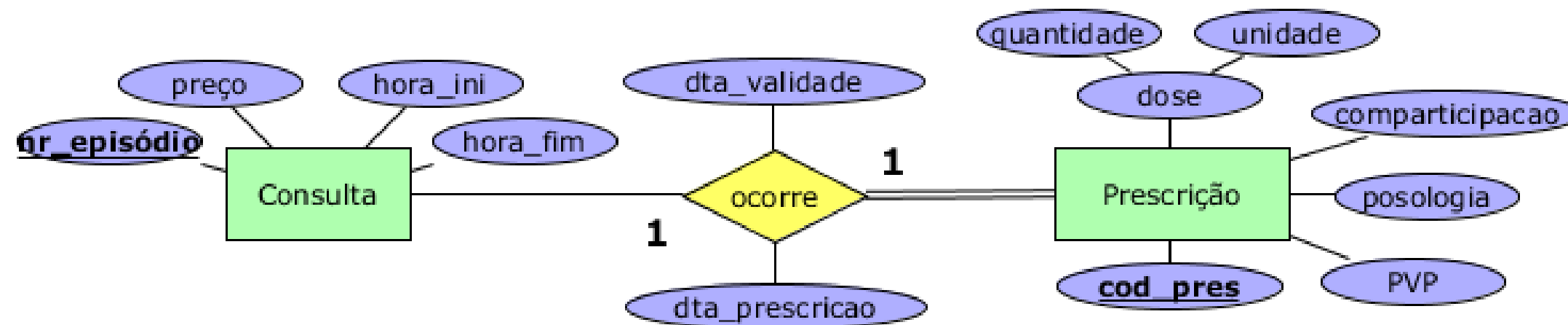
FASE 4: Modelação Lógica

→ Derivar relações

▪ Relacionamentos binários de um-para-um (1:1)

→ (b) participação obrigatória num lado do relacionamento 1:1;

– Cópia da chave primária da entidade pai colocada na relação que representa a entidade filho.



Consulta (nr_episodio, preço, hora_ini, hora_fim)

Chave primária nr_episodio

Prescricao (cod_pres, quantidade, unidade, posologia, PVP, comparticipação, dta_prescrição, dta_validade, nr_episodio)

Chave primária cod_pres

Chave estrangeira nr_episodio **referencia** Consulta(nr_episodio)

FASE 4: Modelação Lógica

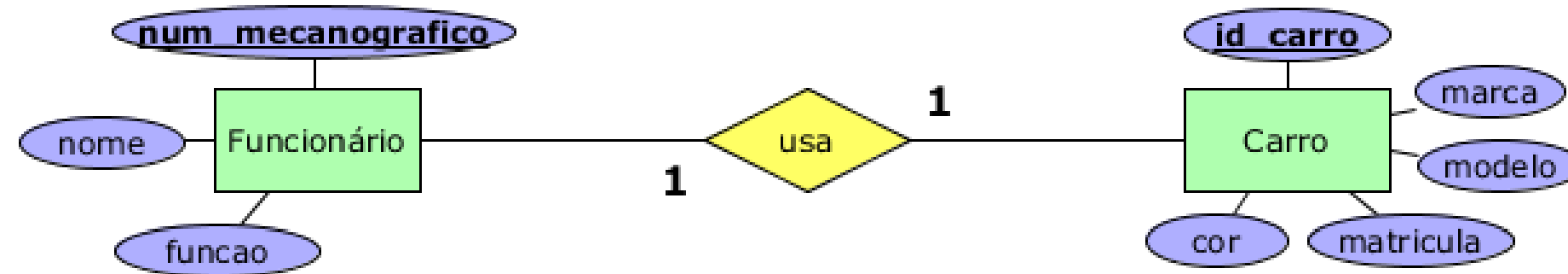
→ Derivar relações

- Relacionamentos binários de um-para-um (1:1)

→ (c) participação opcional em ambos os lados do relacionamento 1:1.

Cópia da chave primária da entidade pai colocada na relação que representa a entidade filho. A designação das entidades pai e filho é arbitrária, a menos que se possa descobrir mais sobre o relacionamento.

EXEMPLO:



Suponha que a maioria dos carros, mas não todos, sejam usados pelos funcionários e que apenas uma minoria dos funcionários use carros. A entidade **Carro**, embora opcional, está mais próxima de ser obrigatória do que a entidade **Funcionário**. Portanto, neste caso deveríamos designar o **Funcionário** como **entidade-pai** e o **Carro** como **entidade-filho**.

FASE 4: Modelação Lógica

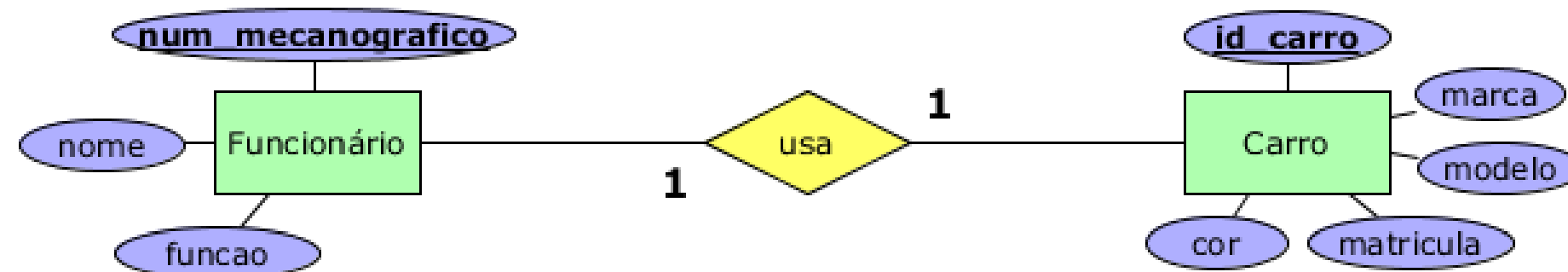
→ Derivar relações

- Relacionamentos binários de um-para-um (1:1)

→ (c) participação opcional em ambos os lados do relacionamento 1:1.

Cópia da chave primária da entidade pai colocada na relação que representa a entidade filho. A designação das entidades pai e filho é arbitrária, a menos que se possa descobrir mais sobre o relacionamento.

EXEMPLO:



Funcionário (num_mecanografico, nome, funcao)
Chave primária num_mecanografico

Carro (id_carro, marca, modelo, matricula, cor,
 num_mecanografico)
Chave primária id_carro
Chave estrangeira num_mecanografico **referencia**
 Funcionário(num_mecanografico)

FASE 4: Modelação Lógica

➔ Derivar relações

Os relacionamentos recursivos de 1:N e N:M seguem as regras de participação de um relacionamento binário de 1:N e N:M, respetivamente.

- Relacionamentos binários recursivos de um-para-um (1:1)

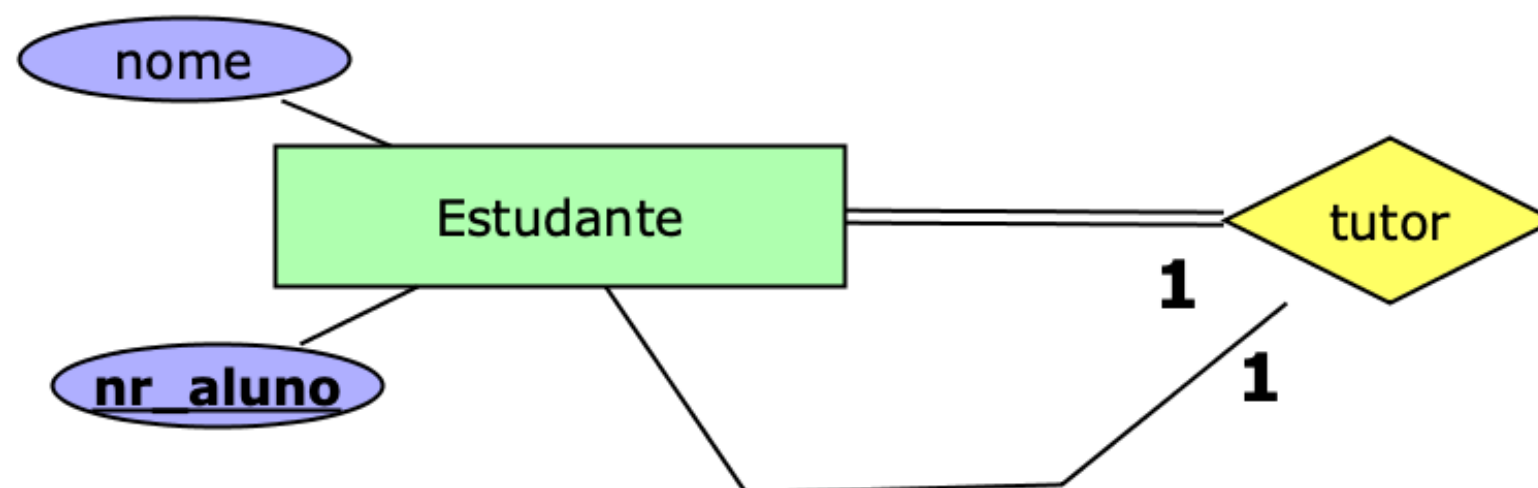
Os relacionamentos recursivos de 1:1 seguem as regras:

- **participação obrigatória de ambos os lados:** relação única com uma cópia da chave primária a agir como chave estrangeira que deve ser renomeada para facilitar a interpretação e não pode ser nula (semelhante ao relacionamento recursivo 1:N).
- **participação opcional de ambos os lados:** criar uma nova relação para representar o relacionamento recursivo que teria apenas dois atributos a funcionar com chave primária composta pelas duas chaves primárias que devem ser renomeadas para facilitar a interpretação e que agem também como chaves estrangeiras (semelhante ao relacionamento recursivo M:N).
- **participação obrigatória em apenas um lado:** opção de seguir qualquer uma das duas abordagens anteriores.

FASE 4: Modelação Lógica

→ Derivar relações

- Relacionamentos binários recursivos de um-para-um (1:1)



Estudante (nr_aluno, nome, tutor)
Chave primária nr_aluno
Chave estrangeira tutor **referencia**
 Estudante(nr_aluno)

ou

Estudante (nr_aluno, nome)
Chave primária nr_aluno

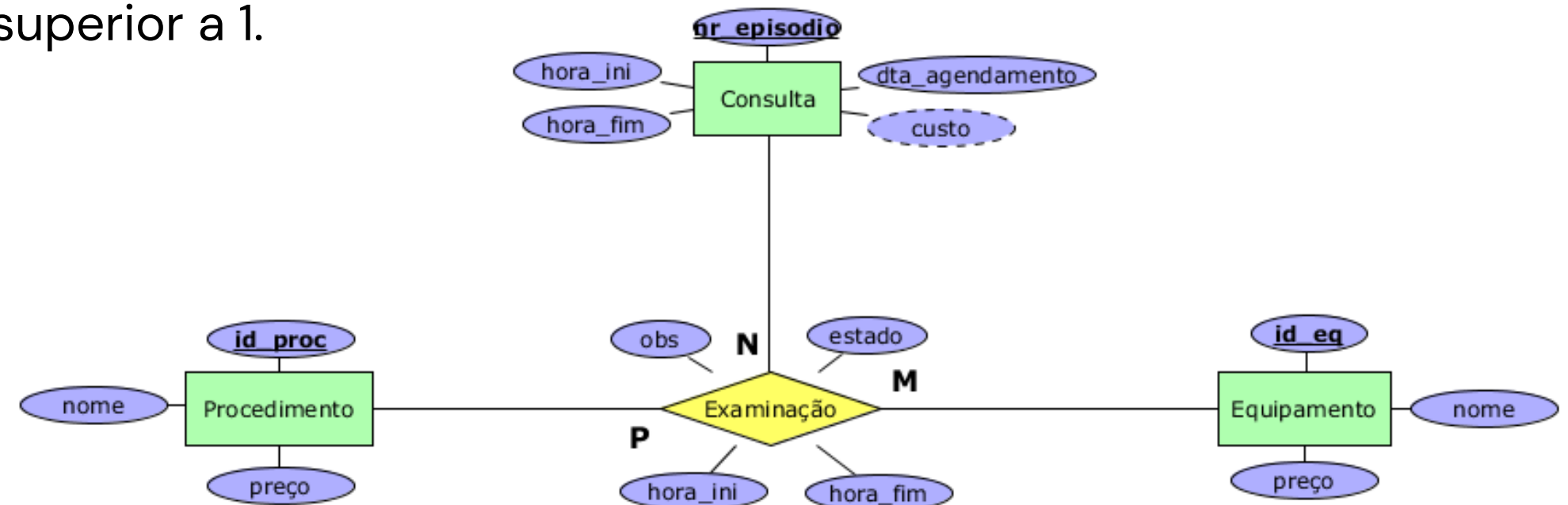
Tutor (nr_aluno, nr_aluno_tutor)
Chave primária nr_aluno,
 nr_aluno_tutor
Chave estrangeira nr_aluno_tutor
referencia Estudante(nr_aluno)
Chave estrangeira nr_aluno
referencia Estudante(nr_aluno)

FASE 4: Modelação Lógica

→ Derivar relações

▪ Relacionamentos complexos

- Para cada relacionamento complexo, criar **uma relação** para representar o **relacionamento** e incluir quaisquer atributos que façam parte do relacionamento.
- Colocamos uma **cópia** da(s) **chave(s) primária(s)** das entidades que participam no relacionamento complexo na nova relação, para atuar como **chaves estrangeiras**.
- A determinação da **chave primária** da nova relação depende da cardinalidade do relacionamento complexo.
- passa a ser composta pelas **chaves primárias** das entidades que participam no relacionamento complexo e que têm cardinalidade superior a 1.



FASE 4: Modelação Lógica

→ Derivar relações

▪ Relacionamentos complexos N:M:P

A nova relação tem uma chave primária composta pelas **chaves primárias** das entidades que participam no relacionamento complexo.

Procedimento (id_proc, nome, preço)

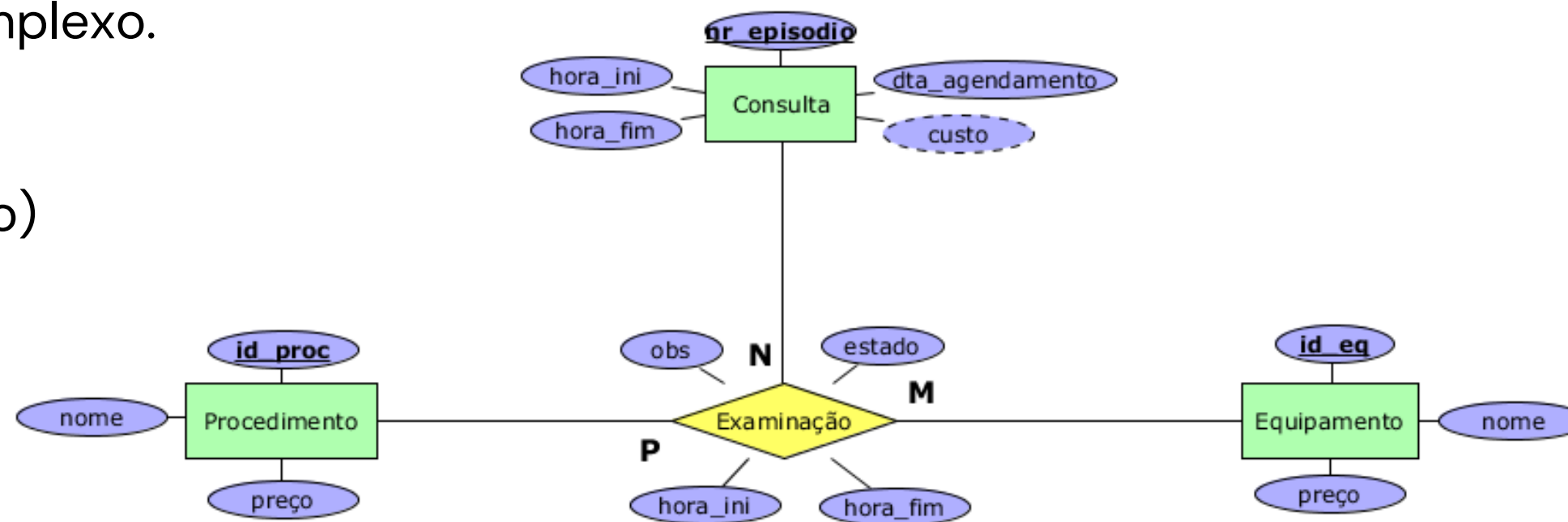
Chave primária id_dproc

Equipamento (id_eq, nome, preço)

Chave primária id_eq

Consulta (nr_episodio, hora_ini, hora_fim, dta_agendamento)

Chave primária nr_episodio



Examinação (nr_episodio, id_proc, id_eq, obs, estado, hora_ini, hora_fim)

Chave primária nr_episodio, id_proc, id_eq

Chave estrangeira nr_episodio **referencia** Consulta(nr_episodio)

Chave estrangeira id_proc **referencia** Procedimento(id_proc)

Chave estrangeira id_eq **referencia** Equipamento(id_eq)

FASE 4: Modelação Lógica

→ Derivar relações

▪ Relacionamentos complexos 1:N:M

A nova relação tem uma chave primária composta pelas **chaves primárias** das entidades que participam no relacionamento complexo com cardinalidade N.

Procedimento (id_proc, nome, preço)

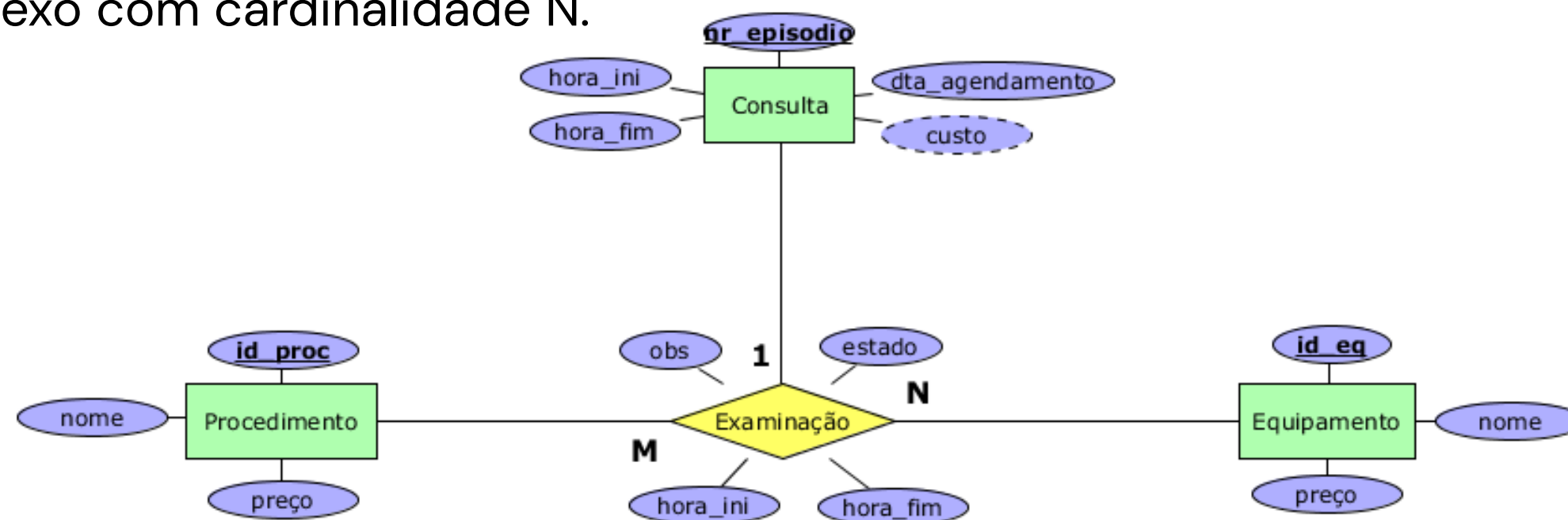
Chave primária id_dproc

Equipamento (id_eq, nome, preço)

Chave primária id_eq

Consulta (nr_episodio, hora_ini, hora_fim, dta_agendamento)

Chave primária nr_episodio



Exame (id_proc, id_eq, nr_episodio, obs, estado, hora_ini, hora_fim)

Chave primária id_proc, id_eq

Chave estrangeira nr_episodio **referencia** Consulta(nr_episodio)

Chave estrangeira id_proc **referencia** Procedimento(id_proc)

Chave estrangeira id_eq **referencia** Equipamento(id_eq)

FASE 4: Modelação Lógica

→ Derivar relações

▪ Relacionamentos complexos 1:1:N

A nova relação tem uma chave primária composta pelas **chave primária** da entidade que participa no relacionamento complexo com cardinalidade N e a chave primária de uma das outras duas entidades, definida de forma arbitrária. Para além disso, o outro par deve ser único.

Procedimento (id_proc, nome, preço)

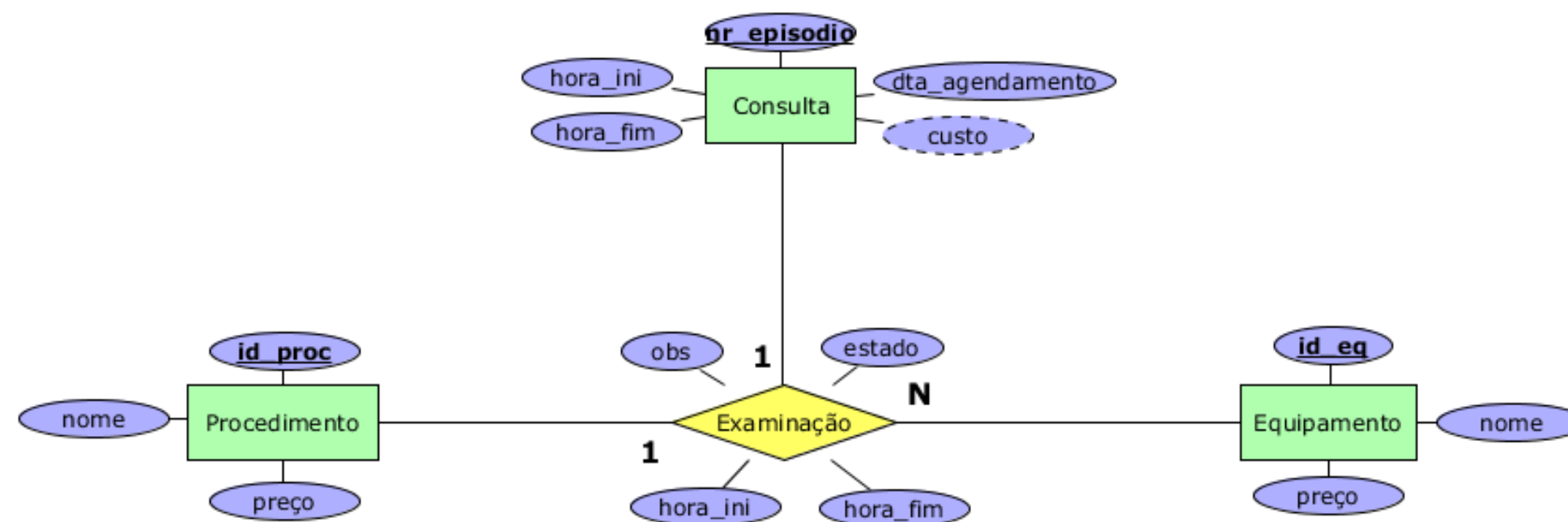
Chave primária id_dproc

Equipamento (id_eq, nome, preço)

Chave primária id_eq

Consulta (nr_episodio, hora_ini, hora_fim, dta_agendamento)

Chave primária nr_episodio



Examinação (id_proc, id_eq, nr_episodio, obs, estado, hora_ini, hora_fim)

Chave primária id_proc, id_eq

Chave estrangeira nr_episodio **referencia** Consulta(nr_episodio)

Chave estrangeira id_proc **referencia** Procedimento(id_proc)

Chave estrangeira id_eq **referencia** Equipamento(id_eq)

FASE 4: Modelação Lógica

→ Derivar relações

▪ Relacionamentos complexos 1:1:1

A nova relação tem uma chave primária composta pelas **chaves primárias** de duas entidades que participam no relacionamento complexo, definidas de forma arbitrária. Para além disso, o outro par deve ser único.

Procedimento (id_proc, nome, preço)

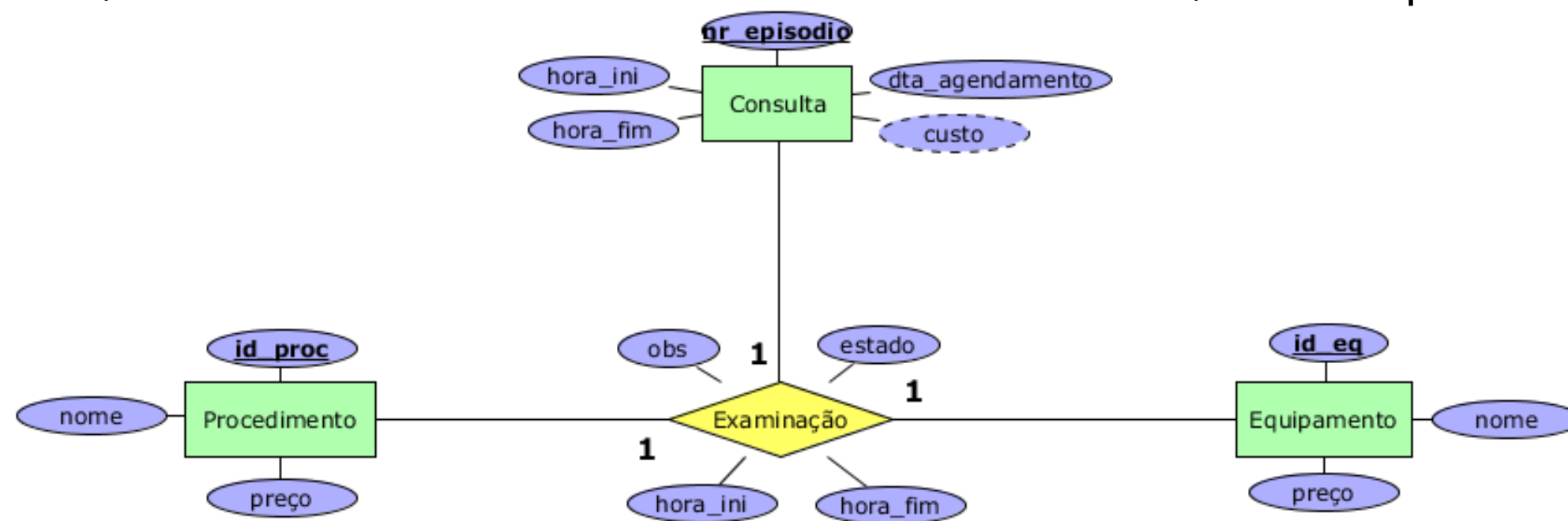
Chave primária id_dproc

Equipamento (id_eq, nome, preço)

Chave primária id_eq

Consulta (nr_episodio, hora_ini, hora_fim, dta_agendamento)

Chave primária nr_episodio



Examinação (id_proc, id_eq, nr_episodio, obs, estado, hora_ini, hora_fim)

Chave primária id_proc, id_eq

Chave estrangeira nr_episodio **referencia** Consulta(nr_episodio)

Chave estrangeira id_proc **referencia** Procedimento(id_proc)

Chave estrangeira id_eq **referencia** Equipamento(id_eq)

FASE 4: Modelação Lógica

→ Derivar relações

- Relacionamentos superclasse/subclasse
 - Identifique a **superclasse** como entidade pai e a **subclasse** como entidade filho.
 - A representação mais adequada de um relacionamento deste tipo depende do número de:
 - restrições de disjunção e participação no relacionamento superclasse/subclasse;
 - se as subclasses estão envolvidas em relacionamentos distintos;
 - número de participantes no relacionamento superclasse/subclasse.

FASE 4: Modelação Lógica

→ Derivar relações

Restrições de Participação	Restrições de Disjunção	Relações Requeridas
Obrigatória	Não disjunto {And}	Relação única com um atributo para cada subclasse (flag)
Opcional	Não disjunto {And}	Duas relações: uma relação para a superclasse e uma relação para todas as subclASSES com um atributo para cada subclasse (flag)
Obrigatória	Disjunto {Or}	Muitas relações (uma relação para cada combinação superclasse/subclasse)
Opcional	Disjunto {Or}	Muitas relações (uma relação para a superclasse e uma para cada subclasse)

FASE 4: Modelação Lógica

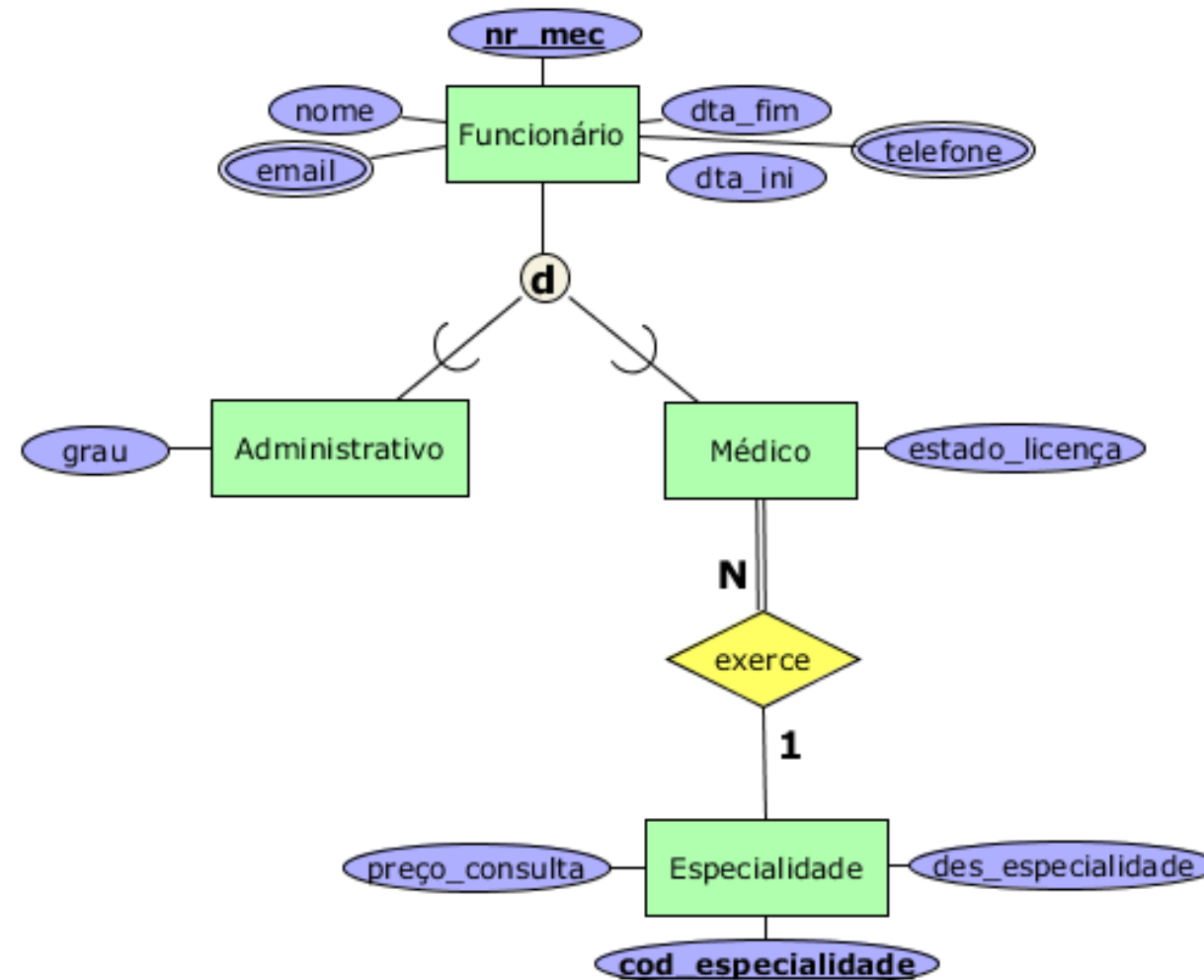
➔ Derivar relações

▪ Relacionamentos superclasse/subclasse

Opcional

Disjunto {Or}

Muitas relações (uma relação para a superclasse e uma para cada subclasse)



FASE 4: Modelação Lógica

→ Derivar relações

▪ Relacionamentos superclasse/subclasse

Funcionário (nr_mecanografico, dta_ini, dta_fim, nome)

Chave primária nr_mecanografico

Médico (nr_mec, estado_licença, especialidade)

Chave primária nr_mec

Chave estrangeira nr_mec **referencia** Funcionario(nr_mec)

Chave estrangeira cod_especialidade **referencia** Especialidade(cod_especialidade)

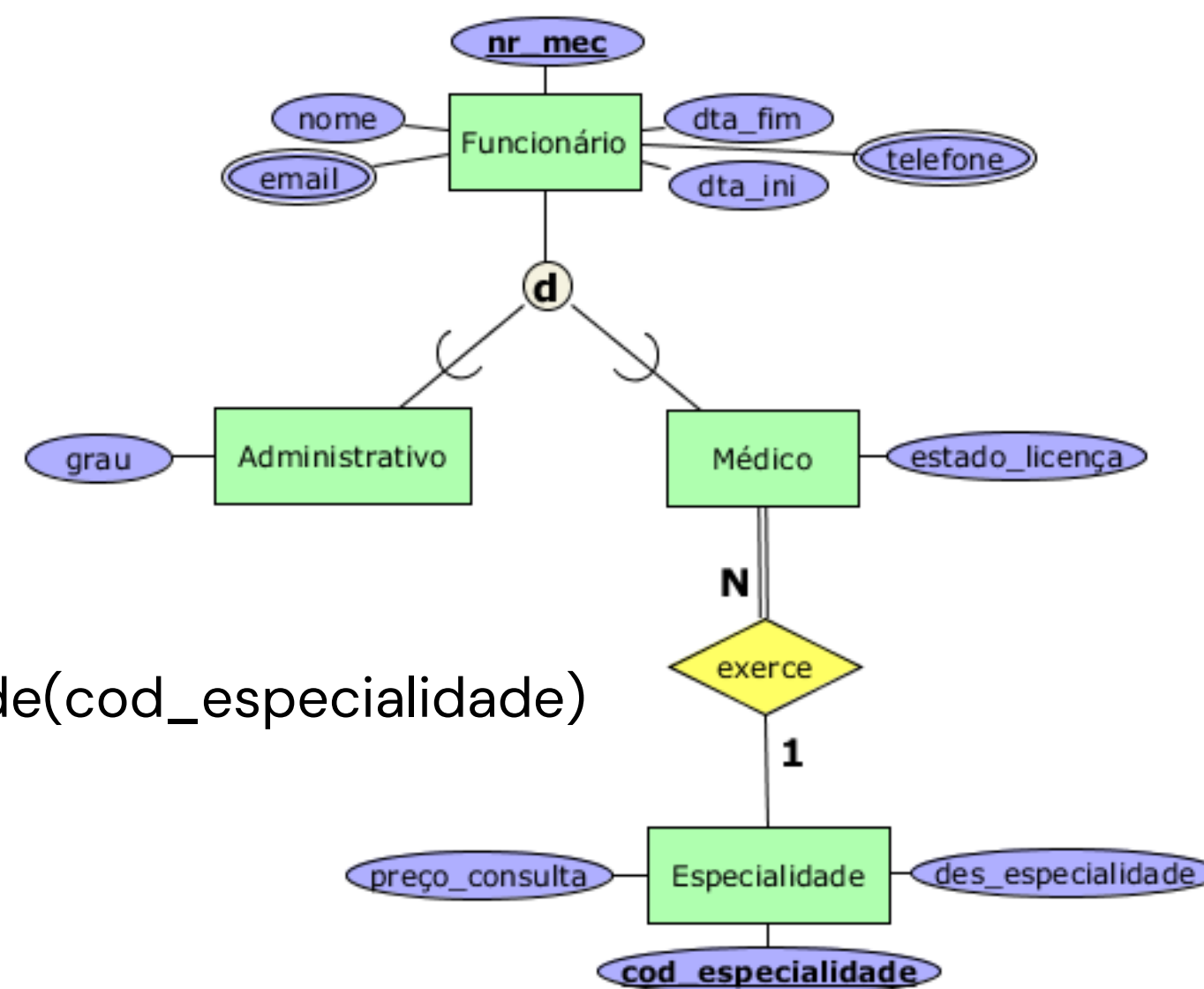
Administrativo (nr_mec, grau)

Chave primária nr_mec

Chave estrangeira nr_mec **referencia** Funcionario(nr_mec)

Especialidade (cod_especialidade, des_especialidade, preço_consulta)

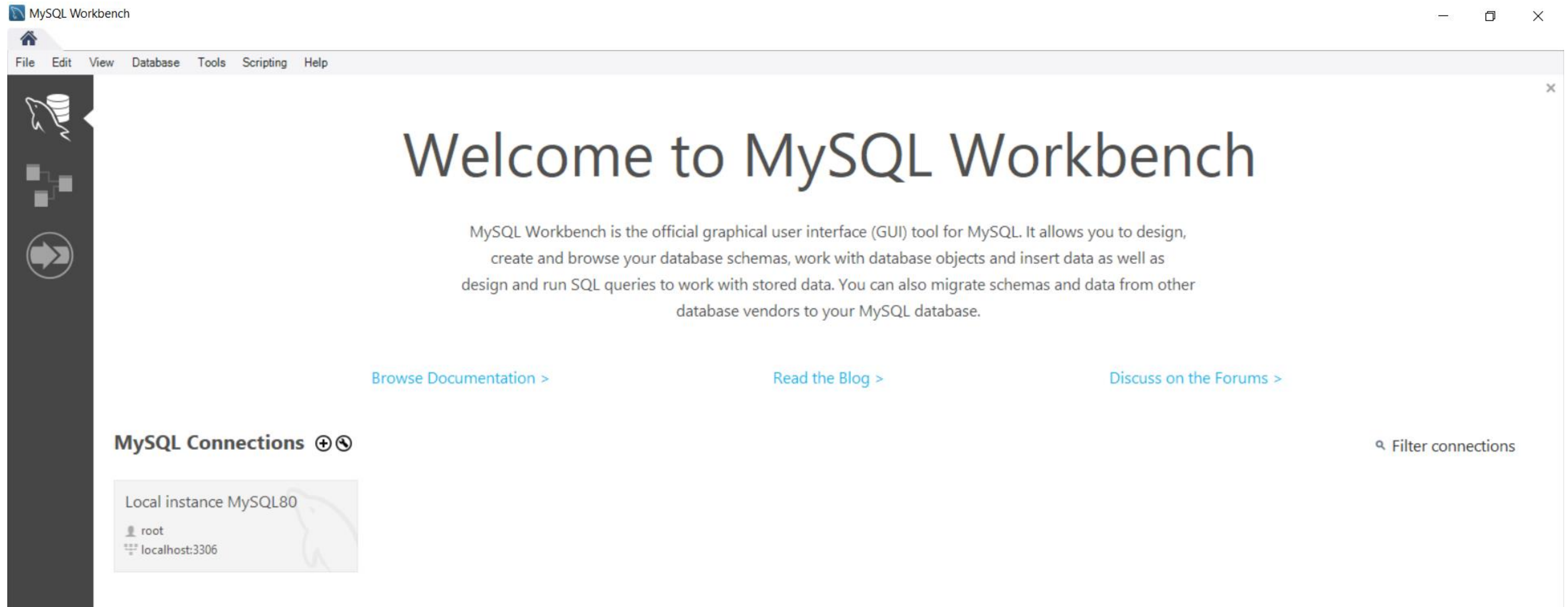
Chave primária cod_especialidade



FASE 4: Modelação Lógica

➔ MySQL Workbench

- 1) Após a instalação, o GUI vai abrir com a configuração ao MySQL server já efetuada (assinalado na figura)

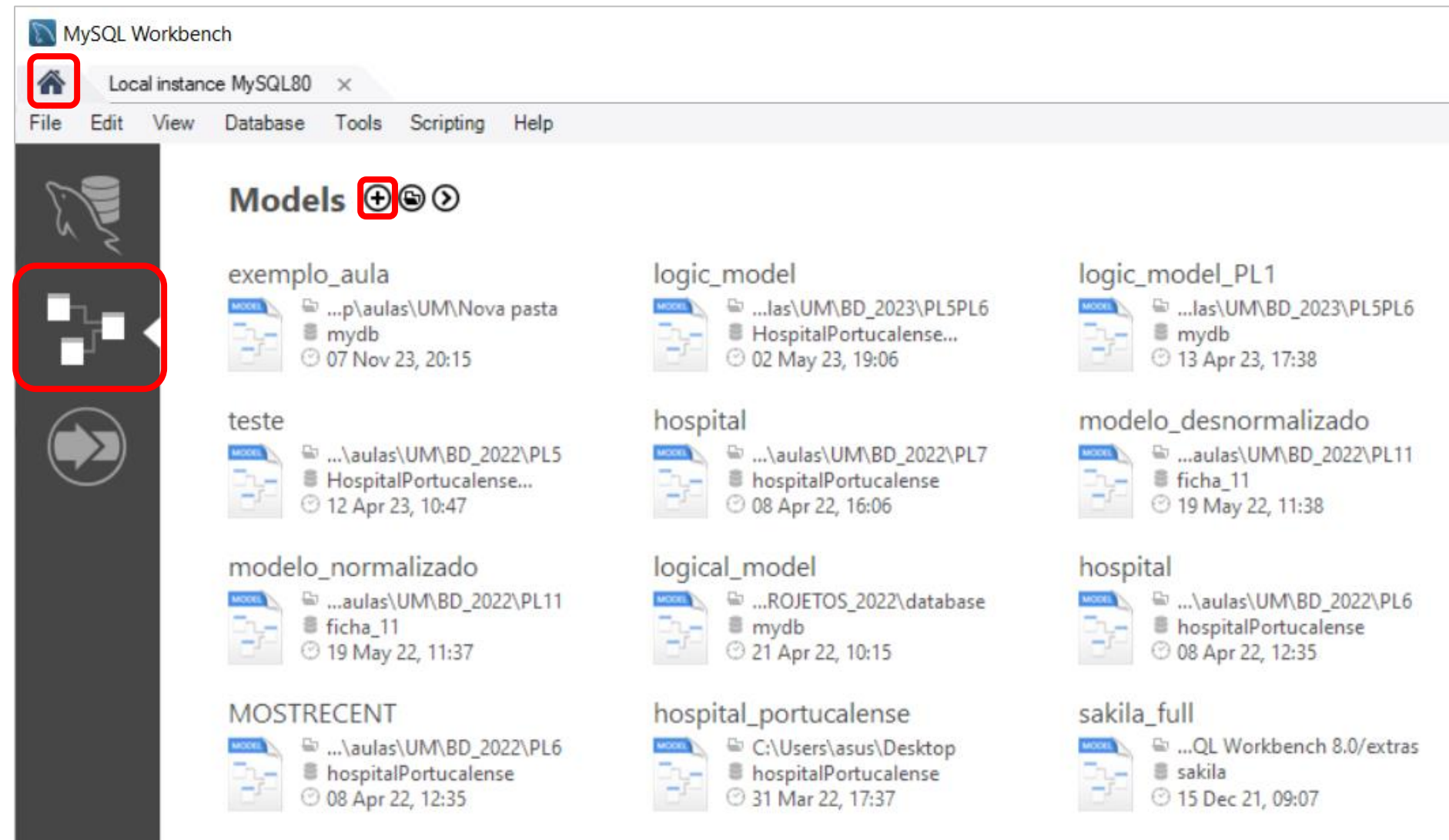


NOTA: se a conexão não aparecer, é provável que falte ou tenha falhado alguma etapa do guia de instalação

FASE 4: Modelação Lógica

➔ MySQL Workbench

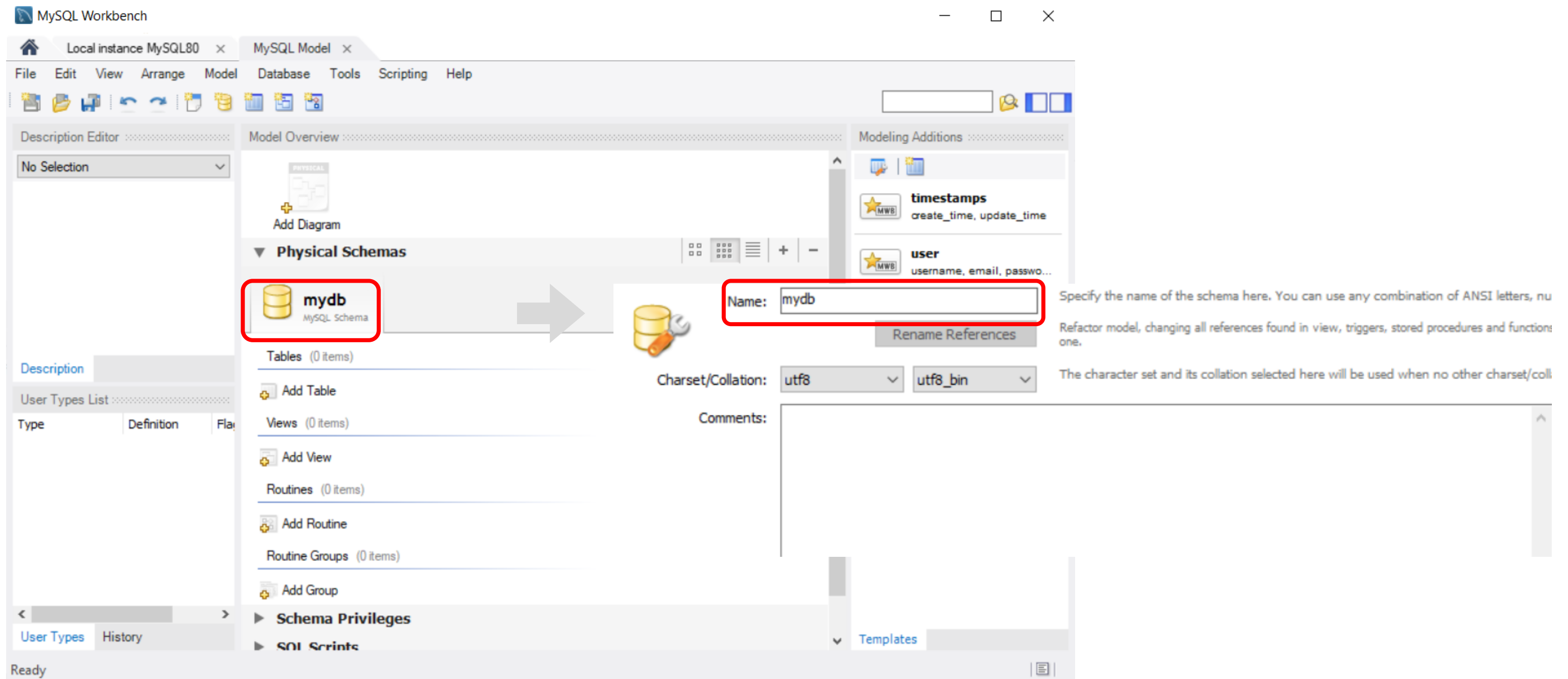
Ir ao menu inicial e clicar no separador “Models” para criar um novo esquema.



FASE 4: Modelação Lógica

➔ MySQL Workbench

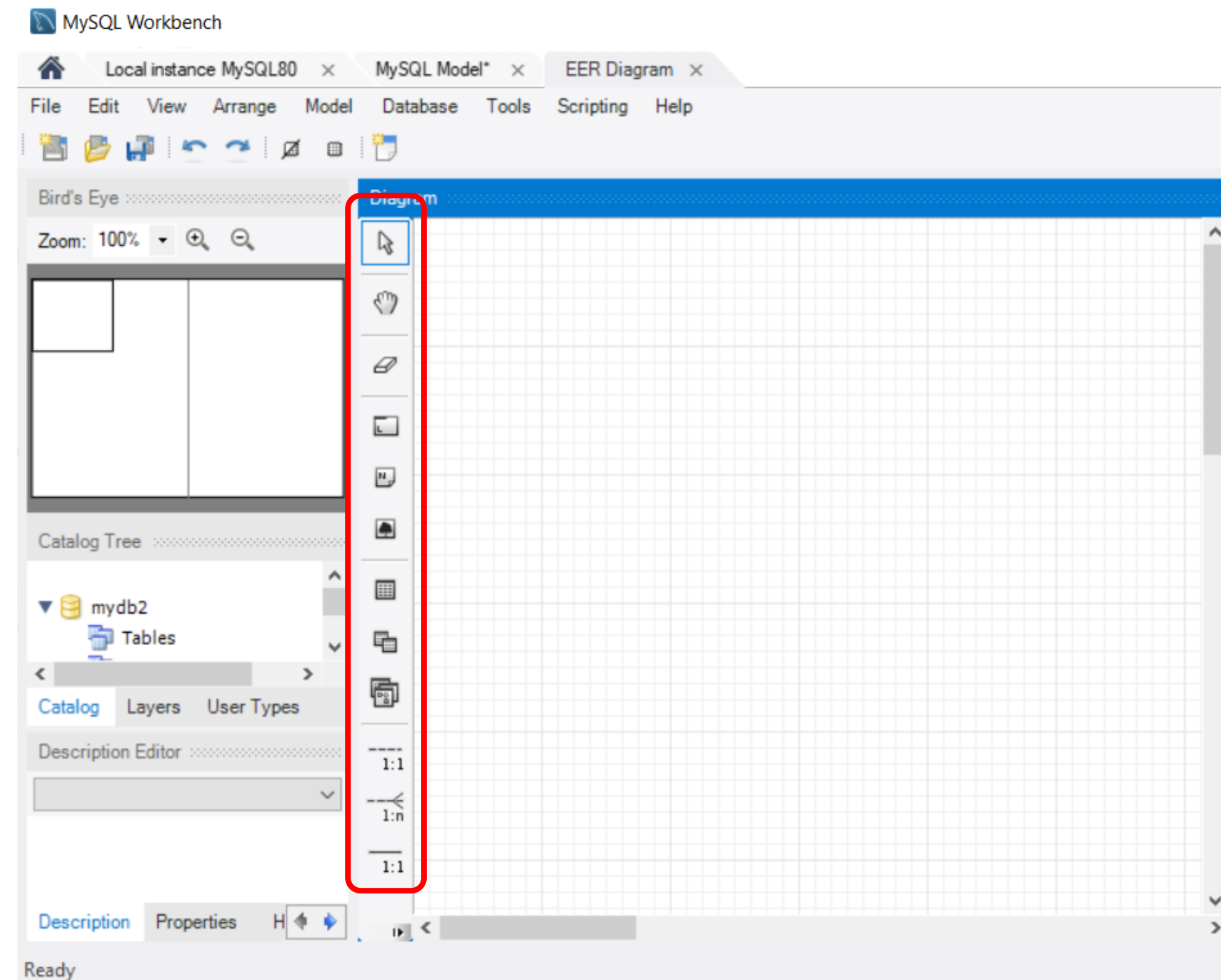
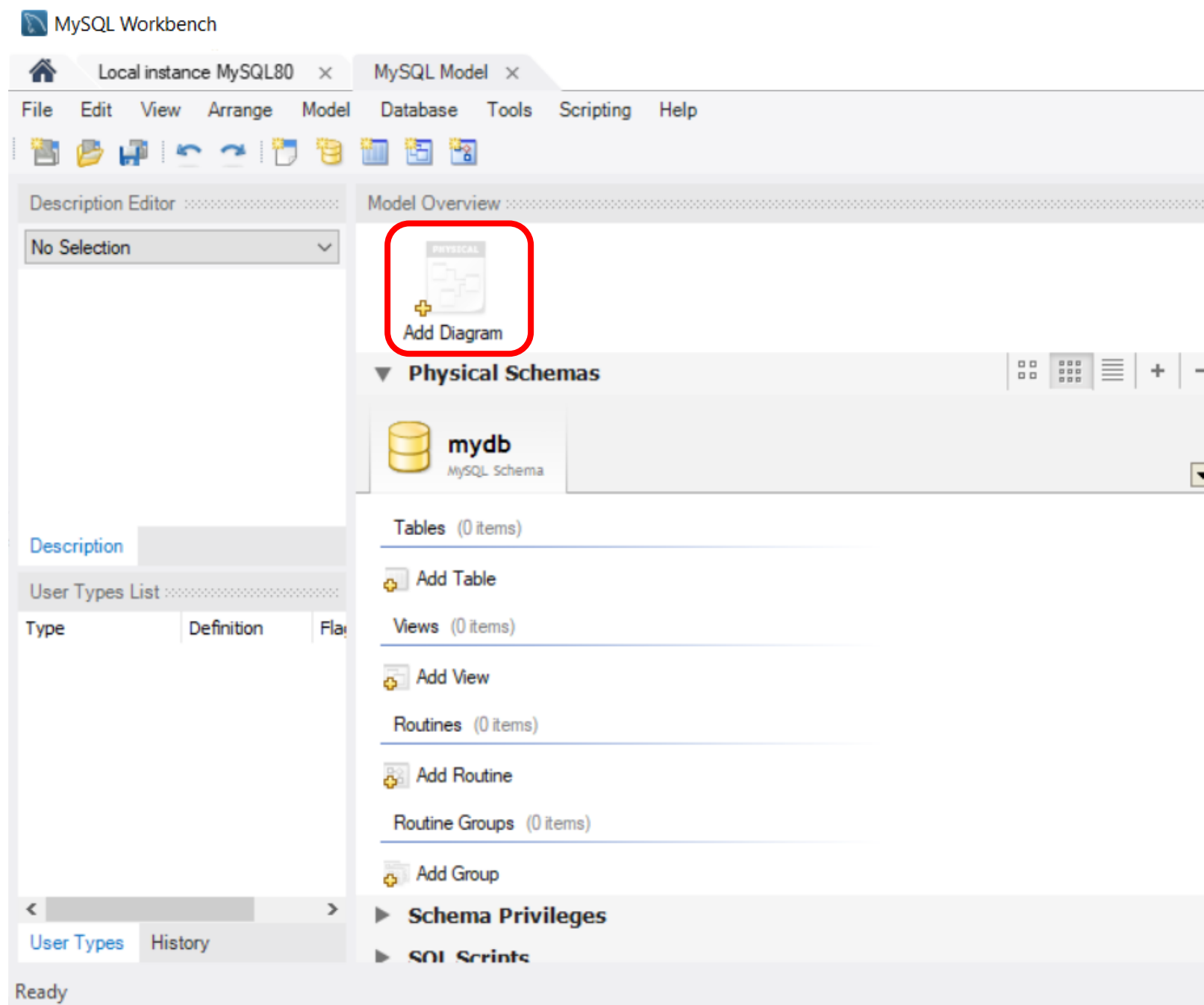
O Workbench cria então um novo esquema com o nome 'mydb'. Para alterar o nome do esquema, basta clicar duas vezes em cima de 'mydb'



FASE 4: Modelação Lógica

➔ MySQL Workbench

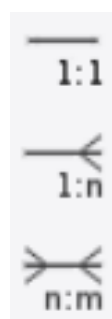
Depois de configurar o nome, clicar no botão 'Add Diagram'. Uma nova janela é criada chamada 'EER Diagram'.



FASE 4: Modelação Lógica – MySQL

Quando estamos a construir o modelo lógico de dados no MySQL, é importante ter em consideração os seguintes aspetos:

- Tipo de relacionamento:



Relacionamentos identificadores (linha cheia) Quando a chave primária da entidade pai é incluída na chave primária da entidade filho.



– Chave estrangeira e chave primária.



Relacionamentos não identificadores (linha tracejada)

Quando a chave primária da entidade pai é incluída na entidade filho, mas não como parte da sua chave primária.



– Chave estrangeira NOT NULL – participação obrigatória no modelo conceptual



– Chave estrangeira – participação opcional no modelo conceptual

- Direcção do relacionamento:

Os relacionamentos devem começar na relação/tabela que deve alocar a chave estrangeira.

FASE 4: Modelação Lógica – MySQL

- Valores padrão/por defeito: Devem ser usados caso se queira considerar um valor por *default*.

Default/Expression

estado_civil	CHAR(1)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'S'
--------------	---------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	-----

Data Type:

Default:

`estado_civil CHAR(1) NULL DEFAULT 'S',`

- PK (Primary Key), NN (Not Null), UQ (Unique Index), B (Binary), UN (Unsigned), ZF (Zero Fill), AI (auto increment), G (generated)
 - PK – deve ser usado para atributos que são chave primária;
 - NN – deve ser usado em todos os atributos de chave primária e todos os atributos que não possam ser NULL;
 - UQ – deve ser aplicado sempre que há chaves candidatas, faz com que não hajam valores duplicados na tabela;
 - UN – define que não podem ser inseridos valores negativos nessa coluna.
 - ZF – preenche o valor definido para o campo com zeros até a largura de exibição especificada na definição da coluna.
 - AI – deve ser usado para gerar automaticamente quando um novo registo é inserido numa tabela.
 - G – deve ser usado para gerar atributos a partir de outros usando uma expressão.

FASE 4: Tipos de Dados no MySQL

➔ Dados Alfanuméricos

<https://dev.mysql.com/doc/refman/8.0/en/data-types.html>

VARCHAR (strings de tamanho **variável**) vs. **CHAR** (strings de tamanho **fixo**)

- O comprimento de dados do tipo CHAR e VARCHAR indica o nº máximo de caracteres que é possível armazenar;
- Os dados do tipo CHAR são preenchidos à direita com **espaços em branco** para o comprimento especificado.

Valor	CHAR(4)		VARCHAR(4)	
"	'____'	4 bytes	"	1 byte
'AB'	'AB__'	4 bytes	'AB'	3 bytes
'ABC'	'ABC_'	4 bytes	'ABC'	4 bytes
'ABCD'	'ABCD'	4 bytes	'ABCD'	5 bytes

O VARCHAR usa 1 ou 2 bytes de memória adicionais para tamanho ou para marcar o fim dos dados.

Para armazenar textos mais longos:

- TEXT
- TINYTEXT
- MEDIUMTEXT
- LONGTEXT

FASE 4: Tipos de Dados no MySQL

➔ Dados Alfanuméricos

<https://dev.mysql.com/doc/refman/8.0/en/data-types.html>

- O tipo **ENUM** é um objeto de string cujo valor é seleccionado a partir de um conjunto de valores permitidos que são definidos explicitamente no momento de criação da coluna.

EXEMPLO:

prioridade ENUM('Não Urgente', 'Pouco Urgente', 'Urgente', 'Muito Urgente', 'Emergente') NOT NULL);

A coluna prioridade aceitará apenas a inserção de um dos cinco valores definidos. O MySQL mapeia cada membro de enumeração para um índice numérico. Neste caso, 'Não Urgente', 'Pouco Urgente', 'Urgente', 'Muito Urgente' e 'Emergente' são mapeados para 1, 2, 3, 4 e 5 respectivamente.

- O tipo **SET** é um objeto string que pode ter zero ou mais valores, cada um dos quais deve ser escolhido a partir de um conjunto de valores especificados quando a tabela é criada.

EXEMPLO:

tipo SET('A', 'B') NOT NULL);

A coluna tipo aceitará a inserção de "", 'A', 'B' ou 'A,B'. O MySQL armazena valores SET numericamente, com o bit de ordem inferior do valor armazenado correspondendo ao primeiro membro do conjunto.

Tipos de Dados no MySQL

➔ Dados de Data/Hora

<https://dev.mysql.com/doc/refman/8.0/en/data-types.html>

Tipo de Dados	Notação
<u>DATE</u>	YYYY-MM-DD
<u>TIME</u>	hh:mm:ss
<u>DATETIME</u> *	YYYY-MM-DD hh:mm:ss
<u>TIMESTAMP</u> **	YYYY-MM-DD hh:mm:ss
<u>YEAR</u>	YYYY

* O intervalo suportado varia de '1000-01-01 00:00:00' a '9999-12-31 23:59:59'.

** O intervalo suportado varia de '1970-01-01 00:00:01' a '2038-01-19 03:14:07'.

Tipos de Dados no MySQL

➔ Dados Numéricos

<https://dev.mysql.com/doc/refman/8.0/en/data-types.html>

- Fixed-Point Types (Exact Value) – DECIMAL

O tipo DECIMAL armazena valores de dados numéricos exatos. Este tipo de dados é usado quando é importante preservar a precisão exata, por exemplo, com dados monetários.

DECIMAL(n,m)

n – precisão – representa o número de dígitos significativos que são armazenados.

m – escala – representa o número de dígitos que podem ser armazenados após o ponto decimal.

Exemplo: 105,98€ → DECIMAL (5,2)

FASE 4: Modelação Lógica

➔ Resolução de Exercícios

Ficha de Exercícios PL05:

Questão 2

Ficha de Exercícios PL06:

Questão 1

FASE 4: Modelação Lógica

→ Restrições de Integridade

Paciente				
<u>nr_sequencial</u>	nome	sexo	dta_nascimento	...
323431	Ana Luísa Dias Gomes	F	20/12/1990	
453347	José da Costa Silva	M	03/05/1975	
212423	Maria Leonor Ribeiro Barbosa	Fem	12/07/2000	
...

✗ Integridade Referencial ?

✗ Integridade de Domínio

✗ Integridade de Entidade

✗ Integridade de Entidade

Consulta							
<u>nr_episodio</u>	<u>id_pac</u>	<u>id_med</u>	hora_ini	hora_fim	id_agenda	cod_proc	id_sec
12345678	212423	3456	2022-01-23 10:18:17	2022-01-23 10:38:27	123456789	P22	1212
14451643	453347	3224	2022-01-25 08:35:23	2022-01-25 09:00:12	223212434	P23	1598
14451643	212423	3371	2022-02-02 09:00:33	2022-02-02 09:15:20	345567811	NULL	1479
13415324	123456	3834	2022-02-04 12:34:11	2022-02-04 13:00:00	433212456	P22	1234
NULL	323431	NULL	2022-02-12 11:20:23	2022-02-12 11:52:33	387612392	P24	1176
...

✗ Exigência de dados

Teoria da Normalização

➔ Normalização de Dados

A normalização de dados baseia-se na análise das **chaves primárias** e das **dependências funcionais** de todos os seus atributos.

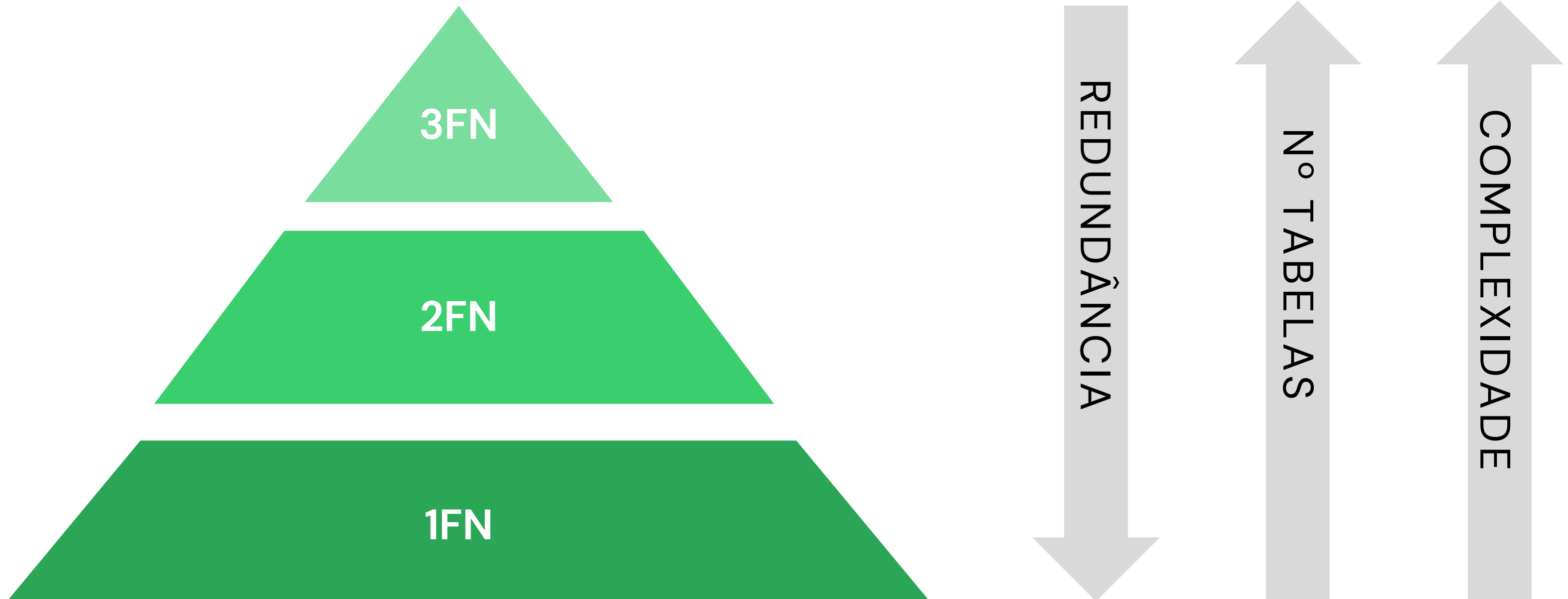
É um processo **progressivo**, que assenta na execução de uma série de etapas, cada uma delas correspondendo a uma **forma normal** específica com critérios de validação cada vez mais fortes.

Através da sua aplicação, os atributos de um dado modelo de dados são organizados para assegurar a **coesão** dos tipos das entidades envolvidas, minimizando ou mesmo **eliminando duplicação de dados**, **melhorando a eficiência de armazenamento**, a **integridade** e a **escalabilidade** dos dados.

Teoria da Normalização

→ Formas Normais

O processo de normalização é **progressivo**, ou seja, cada um dos níveis superiores de normalização é um subconjunto do respetivo nível inferior.



Teoria da Normalização

➔ Primeira Forma Normal – 1FN

Diz-se que uma relação está na 1FN se:

1. Possuir uma chave primária.
2. Todos os seus atributos forem atómicos. Não são permitidos atributos que implicitamente codificam subatributos (atributos compostos) ou atributos multivalor.
3. Não possuir grupos de dados repetitivos.

Na prática, podemos dizer que uma relação está na 1FN se as interseções entre colunas (atributos) e linhas (registos) possuírem um único valor – um valor atómico.

Teoria da Normalização

➔ Primeira Forma Normal – 1FN

Aplicação da 1FN:

Passo 1: Uma das chaves candidatas é escolhida para chave primária.

Passo 2: Atributos multivalor são convertidos em novas relações com chave externa referindo a chave primária da tabela original.

Passo 3: Cada atributo composto é mapeado em vários sub-atributos atômicos.

Teoria da Normalização

➔ Segunda Forma Normal – 2FN

Diz-se que uma relação está na segunda forma normal (2FN) se:

1. A relação estiver também na 1FN.
2. Todos os seus atributos não-primos forem **totalmente dependentes** da sua chave primária. Isto é, não podem existir **dependências parciais**. Diz-se que um atributo é não-primo quando este não faz parte de uma chave primária.

Teoria da Normalização

➔ Segunda Forma Normal – 2FN

O que é uma dependência funcional?

As dependências funcionais determinam a forma como se pode interpretar e relacionar os dados e permitem especificar medidas formais sobre a correção dos esquemas relacionais.

Na prática, a dependência funcional $A1 \rightarrow A2$ entre dois conjuntos de atributos de uma relação significa que:

- para cada valor de A1, existe apenas um valor possível para A2, por isso diz-se que A2 é funcionalmente dependente de A1 ou que A1 determina funcionalmente A2;
- valores iguais para A1, determinam valores iguais para A2.

Teoria da Normalização

➔ Segunda Forma Normal – 2FN

Aplicando a análise de dependências funcionais:

Alunos(id_aluno, nome_aluno, cod_curso, nome_curso)

Notas(id_aluno, cod_dis, nome_dis, nota, cod_prof, nome_prof, dep_prof)

Diagrama de Dependências

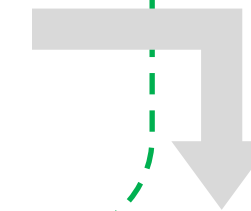
id_aluno → nome_aluno, cod_curso

cod_curso → nome_curso

{id_aluno, cod_dis} → nota

cod_disc → **nome_dis, cod_prof**

cod_prof → nome_prof, dep_prof



A tabela Alunos está na 2FN, mas a Notas não!

Teoria da Normalização

➔ Segunda Forma Normal – 2FN

Diagrama de Dependências

$\{id_aluno, cod_dis\} \rightarrow nota$

$cod_disc \rightarrow nome_dis, cod_prof$

$cod_prof \rightarrow nome_prof, dep_prof$

Notas(id_aluno, cod_dis, nome_dis, nota, cod_prof, nome_prof, dep_prof)

<u>id_aluno</u>	<u>cod_dis</u>	nome_dis	nota	cod_prof	nome_prof	dep_prof
001	D01	Bases de Dados	16	P01	Maria do Carmo	Dep. Informática
001	D02	Criptografia	12	P02	Paulo Gomes	Dep. Informática
002	D01	Bases de Dados	17	P01	Maria do Carmo	Dep. Informática
002	D03	Lógica Computacional	14	P03	Tiago Pinho	Dep. Sistemas

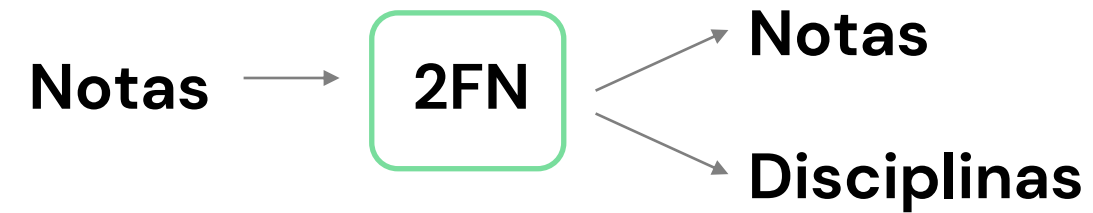
... ..



Dependência Parcial

Teoria da Normalização

➔ Segunda Forma Normal – 2FN



Notas
<u>id_aluno</u>
<u>cod_dis</u>
nota

(001, 'D01', 16)
(001, 'D02', 12)
(002, 'D01', 17)
(002, 'D03', 14)
...

Disciplinas(cod_dis, nome_dis, cod_prof, nome_prof, dep_prof)

Disciplinas
<u>cod_dis</u>
nome_dis
cod_prof
nome_prof
dep_prof

Teoria da Normalização

➔ Terceira Forma Normal – 3FN

Diz-se que uma relação está na 3FN se:

1. A relação estiver também na 1FN e na 2FN.
2. Todos os seus atributos que não sejam chaves primárias sejam mutuamente independentes, não havendo assim **dependências funcionais transitivas**. Por outras palavras, numa relação na 3FN, todos os atributos dependem única e exclusivamente da chave primária.

Na prática, isto significa que os atributos que não dependam da chave primária devem ser “eliminados” da relação, ou seja, devem ser transferidos para outra tabela.

Teoria da Normalização

→ Terceira Forma Normal – 3FN

Na prática, a dependência funcional $A1 \rightarrow A2, A3$ e $A3 \rightarrow A4$ entre os atributos de uma relação significa que:

- Existe uma dependência funcional transitiva entre $A1$ e $A4$. Ou seja os atributos que não são chave primária, não são mutuamente independentes entre si.

Aplicando a análise de dependências funcionais ao caso de estudo anterior:

Diagrama de Dependências

Alunos(id_aluno, nome_aluno, cod_curso, nome_curso)

Notas(id_aluno, cod_dis, nota)

Disciplinas(cod_dis, nome_dis, cod_prof, nome_prof, dep_prof)

$id_aluno \rightarrow nome_aluno, cod_curso$

$cod_curso \rightarrow nome_curso$

$\{id_aluno, cod_dis\} \rightarrow nota$

$cod_disc \rightarrow nome_dis, cod_prof$

$cod_prof \rightarrow nome_prof, dep_prof$

A tabela Alunos e Disciplinas não estão na 3FN!

Teoria da Normalização

→ Terceira Forma Normal – 3FN

Alunos(id_aluno, nome_aluno, cod_curso, nome_curso)

<u>id_aluno</u>	<u>nome_aluno</u>	<u>cod_curso</u>	<u>nome_curso</u>
001	João Ferreira	C01	MIEI
001	João Ferreira	C01	MIEI
002	Rita Abreu	C01	MIEI
002	Rita Abreu	C01	MIEI
...

Diagrama de Dependências

id_aluno → nome_aluno, cod_curso
cod_curso → nome_curso

Dependência Transitiva

Teoria da Normalização

→ Terceira Forma Normal – 3FN

Diagrama de Dependências

$\text{cod_disc} \rightarrow \text{nome_dis}, \text{cod_prof}$
 $\text{cod_prof} \rightarrow \text{nome_prof}, \text{dep_prof}$

Disciplinas(cod_dis, nome_dis, cod_prof, nome_prof, dep_prof)

<u>cod_dis</u>	nome_dis	cod_prof	nome_prof	dep_prof
D01	Bases de Dados	P01	Maria do Carmo	Dep. Informática
D02	Criptografia	P02	Paulo Gomes	Dep. Informática
D01	Bases de Dados	P01	Maria do Carmo	Dep. Informática
D03	Lógica Computacional	P03	Tiago Pinho	Dep. Sistemas

...

...

...

...

...

Dependência Transitiva

Teoria da Normalização

→ Terceira Forma Normal – 3FN



Alunos(id_aluno, nome_aluno, cod_curso)

Alunos
<u>id_aluno</u>
nome_aluno
cod_curso

Cursos(cod_curso, nome_curso)

Cursos
<u>cod_curso</u>
nome_curso



Disciplinas(cod_dis, nome_dis, cod_prof)

Disciplinas
<u>cod_dis</u>
nome_dis
cod_prof

Professores(cod_prof, nome_prof, dep_prof)

Professores
<u>cod_prof</u>
nome_prof
dep_prof

Teoria da Normalização

→ Terceira Forma Normal – 3FN

Alunos(id_aluno, nome_aluno, cod_curso)

Alunos
<u>id_aluno</u>
nome_aluno
cod_curso

Cursos(cod_curso, nome_curso)

Cursos
<u>cod_curso</u>
nome_curso

Diagrama de Dependências

$\text{id_aluno} \rightarrow \text{nome_aluno}, \text{cod_curso}$
 $\text{cod_curso} \rightarrow \text{nome_curso}$
 $\{\text{id_aluno}, \text{cod_dis}\} \rightarrow \text{nota}$
 $\text{cod_disc} \rightarrow \text{nome_dis}, \text{cod_prof}$
 $\text{cod_prof} \rightarrow \text{nome_prof}, \text{dep_prof}$

Disciplinas

Disciplinas
<u>cod_dis</u>
nome_dis
cod_prof

Disciplinas

Disciplinas
<u>cod_prof</u>
nome_prof
dep_prof

As relações encontram-se na 3FN!