

Universidade de São Paulo

ICMC – Instituto de Ciências Matemáticas e de Computação

SSC0640 - Bases de Dados

Prof. Dra. Elaine Parros M. de Sousa
PAE: André Moreira Souza

REDE SIS-SAÚDE

UMA SOLUÇÃO PARA GESTÃO DE RECURSOS

Caio Florentin de Oliviera – 14562921

Enrico Martim Zúollo – 14568048

Guilherme Sonego – 14576489

Pedro Arthur D. E. S. Sanches – 5015792

São Carlos - SP
2025

Sumário

1	Introdução	2
2	Modelo Entidade-Relacionamento	2
2.1	Levantamento de Requisitos	2
2.2	Principais Funcionalidades	4
2.3	Restrições de integridade	5
2.4	Diagrama do Modelo Entidade-Relacionamento	6
2.5	Mudanças relacionadas à Primeira Entrega	8
3	Modelo Relacional	9
3.1	Mapeamentos das Entidades e Generalizações	11
3.2	Mapeamentos dos Relacionamentos	12
3.3	Mapeamentos das Agregações	14
3.4	Mapeamentos dos atributos	16
3.5	Mudanças relacionadas à Segunda Entrega	17
4	Implementação	18
4.1	Aplicação	18
4.2	Script de consultas	19
5	Conclusão	23

1 Introdução

O sistema de saúde brasileiro é reconhecido como um dos maiores e mais abrangentes do mundo, garantindo cobertura universal à população. No entanto, ainda apresenta falhas significativas relacionadas à **gestão de recursos**, como o **desperdício** de mais de **R\$ 2 bilhões** em medicamentos e vacinas vencidas desde 2019, conforme apontado pela reportagem (ABDALA, 2023). Além disso, esse sistema é marcado pela **desigualdade regional** no acesso a recursos, especialmente nas regiões Norte e Nordeste, situação evidenciada em estudo conduzido pelo Centro de Integração de Dados e Conhecimentos para Saúde (Cidacs/Fiocruz Bahia) (ANJOS, 2022).

Com base nesses problemas, nosso grupo decidiu propor uma solução para evitar o desperdício de recursos dentro das entidades públicas de saúde e diminuir a desigualdade de acessos a estes. A proposta consiste no desenvolvimento de um sistema capaz de **unificar os estoques** das diferentes unidades de saúde, possibilitando a **visualização centralizada** da disponibilidade de insumos e leitos. Dessa forma, diante da falta de determinado recurso em uma unidade, será possível solicitar sua **transferência** de outra instituição que o possua em estoque, promovendo maior eficiência na gestão, evitando perdas e garantindo um atendimento mais equitativo à população.

Adicionalmente, o sistema proposto contempla uma funcionalidade de **análise de tendências** e previsão de possíveis faltas de recursos. Para tal, prevê-se a implementação de um módulo no qual os profissionais de saúde poderão registrar os casos atendidos, bem como os insumos necessários para o tratamento de cada paciente (por meio de relatórios). Esses registros fornecerão dados consistentes para a identificação de **padrões de consumo, detecção de surtos locais e antecipação de demandas específicas**. A partir dessas análises, será possível planejar e solicitar de forma proativa a redistribuição ou aquisição de recursos nas regiões mais afetadas, promovendo uma gestão mais eficiente e preventiva.

2 Modelo Entidade-Relacionamento

2.1 Levantamento de Requisitos

Uma **entidade de saúde** pode ser, dentre outras possibilidades, um **laboratório** ou um **hospital**. Essas são as que mais nos interessam dentro do nosso sistema. Todas possuem, porém, um **endereço, horário de funcionamento, telefone, nome** e um **CNES** (Cadastro Nacional de Estabelecimentos de Saúde) que a identifica.

Com relação às **especializações** de **entidades de saúde**, os **laboratórios** são **entidades de saúdes** que possuem estoques de **recursos** (medicamentos/vacinas) sendo responsáveis também por sua **produção**. Cada **laboratório** pode ser responsável pela produção de um ou mais **recursos**, além de **receberem e processarem** requisições de **recursos** vindos de outras unidades.

Já os **hospitais** também têm estoques de **recursos** (medicamentos/vacinas), podendo receber e processar requisições de **recursos**. Porém, diferentemente dos **laboratórios**, os **hospitais** não produzem **recursos**, mas possuem **leitos**, a partir dos quais podem ainda receber e encaminhar **pacientes** para outras **unidades de saúde**, de acordo com a disponibilidade de **leitos** e as **especialidades** daquele **hospital** (relacionado às cirurgias que podem ser realizadas naquele local). É importante perceber que não podemos encaminhar um paciente para a mesma entidade de saúde que ele se encontra (Nota 3 do diagrama).

Dentro das **entidades de saúde** atuam diferentes tipos de profissionais (**funcionários**), os quais são contratados para trabalhar em uma **escala de trabalho** com os **dias da semana** e **horários** definidos, assim como uma **função** (o qual pode ser **médico** ou **enfermeiro** –

usuários do sistema). Porém, o trabalho de fato é contado a partir de um **turno** daquele trabalhador, isto é, do **momento** em que ele **chega** até o momento em que ele **sai** da entidade de saúde. É preciso garantir que um **funcionário** não esteja associado a **turnos** que ocorram no mesmo dia e horário em unidades diferentes (Nota 2 do diagrama).

É durante um **turno** que o trabalhador pode usar os recursos em estoque da entidade que trabalha, **atender os pacientes** e **solicitar recursos** faltantes do estoque para atender melhor os pacientes. Eles também podem querer **encaminhar pacientes** para outras **entidades de saúde** quando não há **leitos disponíveis** ou o tratamento não é possível naquela unidade, emitindo assim um **relatórios de caso**.

Estes **relatórios de casos** são o resultado do atendimento de um **funcionário** com um **paciente** em um **turno** de uma **entidade de saúde**, sendo que eles podem conter informações quanto às **doenças** que um **paciente** apresenta ou uma **necessidade de encaminhamento**. Ambas essas informações do paciente são de utilidade para o **gestor do sistema** – outro usuário do sistema. Além disso, o **relatório** possui um campo para uma (ou mais) **palavras-chave**, que pode ser, por exemplo, o nome de uma doença, vírus que o **paciente** apresenta, o que ajuda o **gestor** a encontrar padrões em determinada região. Devemos assegurar que os **timestamps** dos **relatórios** sejam consistentes com o **turno** em que foram gerados (Nota sobre Timestamp do diagrama).

Os **pedidos** de recursos, assim como os relatórios de casos, também são vinculados a um **turno**, e consequentemente, a um **funcionário**; neles há um **recurso** atrelado, contendo informações como **quantidade**, **urgência** e **momento do pedido** (timestamp). Esses **pedidos** são **analizados** pelos **diretores das entidades**, os quais são responsáveis por **gerar relatórios de recurso**, como um filtro e formalização para os **pedidos**.

Os **relatórios de recursos** documentam, além das informações contidas no pedido, o **diretor** que o analisou, a **urgência** do pedido (reinterpretada pelo diretor), o **momento** em que o **relatório** foi gerado (**timestamp**), o parecer do diretor para o pedido (negado/aceito), referenciado pelo **estado**, e um breve **texto** de explicação/contextualização do **pedido** e do parecer do **diretor**. Depois de gerado o **relatório do pedido**, ele pode ser **aprovado** ou não pelo **gestor do sistema**. Caso **aprovada**, gera-se uma **requisição**, em que uma ou mais **transportadoras** são associadas a tal **requisição**, possibilitando a movimentação dos **recursos**.

Cada um dos **recursos** tem seu próprio **código da ANVISA** (Agência Nacional de Vigilância Sanitária), **nome comercial** e **tipo** (remédio, vacina, etc). Além disso, possui um campo para **temperatura de armazenamento**, a qual é usada para escolher a **transportadora** adequada para movimentação do **recurso**. Essa **transportadora** é cadastrada no banco com o **CNPJ**, **nome** e **telefone**. Essas empresas podem ser associadas a diferentes **hospitais** e têm a responsabilidade de levar os **recursos** solicitados às unidades geradoras do **pedido**. Um mesmo **recurso** pode, inclusive, ser transportado por diferentes **transportadoras** a partir de **hospitais** distintos, dependendo da logística. Para ser contratada em dada **requisição**, porém, ela deve possuir o transporte adequado para uma carga que deve ser armazenada a uma **temperatura específica**. O resultado de transportar o **recurso** gera um **tempo de rota**.

Por fim, vale salientar que a necessidade de formalização das ações contempladas pelos **relatórios de casos e recursos** são de utilidade para o **gestor do sistema**, o qual atua em nível superior: da análise dos **relatórios de casos**, o **gestor** pode encaminhar ou não um **paciente** para outra entidade, devido a uma possível demanda alta e prioridades de pacientes. Além disso, a análise pode ser utilizada também para identificar tendências epidemiológicas, como o surgimento de surtos ou ondas de determinadas doenças em uma região, podendo gerar **notificações** para outras **entidades**. Da análise dos **relatórios de recursos** ele pode gerar **notificações** quanto à falta de um dado **recurso** na região. Essas **notificações** contém o campo de **texto**, para que seja descrito o problema e o momento em que foi mandada.

2.2 Principais Funcionalidades

As operações realizáveis no Banco de Dados estão relacionadas aos trabalhadores de uma Entidade de Saúde, ao gestor do sistema e a um paciente.

Enfermeiro

- Inserir, atualizar e remover itens do estoque da entidade de saúde que trabalha.
- Buscar por itens em estoque.
- Solicitar recursos que não estão em estoque.

Médico

- Inserir, atualizar e remover itens do estoque da entidade de saúde que trabalha.
- Buscar por itens em estoque.
- Solicitar recursos que não estão em estoque.
- Gerar relatórios de casos atrelados a um paciente.
- Encaminhar pacientes para outras entidades de saúde.
- Consultar todos os trabalhadores que estão trabalhando naquele turno.
- Consultar todos os relatórios gerados por ele (em qualquer turno e entidade que trabalhe).

Diretor

- Cadastrar novos trabalhadores para a entidade.
- Visualizar todos os trabalhadores da entidade e os que estão trabalhando no momento.
- Buscar por recursos no estoque da entidade que trabalha.
- Visualizar e Analisar pedidos relacionados a sua entidade e gerar relatórios de recursos.

Gestor

- Cadastrar, atualizar, remover e buscar pelas transportadoras no sistema.
- Visualizar todos os relatórios de casos e recursos no sistema, podendo fazer análises estatísticas para prever tendências.
- Notificar entidades de saúde quanto a possibilidade de surtos e falta de recursos.
- Cadastrar novas entidades de saúde e trabalhadores para elas.
- Buscar o estoque de um dado recurso em todas as entidades da região.
- Buscar laboratórios que produzem um dado recurso.
- Aprovar o envio de um recursos de uma entidade para outra (requisições).
- Aprovar a transferência de um paciente entre entidades.

Paciente

- Visualizar o relatório atrelado a seu caso.

2.3 Restrições de integridade

Ciclo: Laboratório → Recurso:

Trata-se de um ciclo envolvendo as relações "**possui**", atrelado a uma **entidade de saúde**, e "**produz**", exclusivo dos **laboratórios**. Esse ciclo é natural de nossa modelagem, uma vez que são ações distintas. Um **laboratório** pode produzir um **recurso** que não possui e pode possuir um **recurso** que não **produz**, mas o fato do **laboratório** produzir o **recurso** pode ajudar um **gestor** em casos que o recurso está escasso na região como um todo.

Ciclo: Entidade de Saúde → Turno → Pedido → Recurso

Esse ciclo envolve as relações "**possui**"/"**produz**", "**pede**" e o agregado "**turno**" da relação "**trabalha**". Trata-se, porém, de um ciclo natural da nossa modelagem e representa a situação em que um **funcionário**, durante um **turno** de uma **entidade de saúde**, realiza um pedido de dado recurso, que uma entidade de saúde possui. Esse relacionamento não deve ser considerado um problema, pois o **funcionário** pode também pedir recursos já disponíveis no estoque da entidade em que trabalha, pois estes podem estar em quantidades inferiores à recomendada, ou ainda, este pedido pode também se basear em uma notificação do **gestor** pedindo que aumente o estoque; cabe, porém, ao **diretor analisar** se este pedido é válido nesses casos.

Ciclo: Diretor → Entidade de Saúde → Turno → Pedido

Este ciclo pode ser problemático dentro da aplicação por gerar um **ciclo de dependência** e envolve as relações "**trabalha**" e "**analisa**". Como a função do **diretor** é analisar os pedidos feitos na entidade de saúde em que trabalha, é necessário garantir que ele analise apenas os pedidos realizados nas entidades de saúde em que trabalha. Vamos ter que tratar disso na aplicação.

Ciclo: Hospital → Turno → Relatório de Caso

Este ciclo envolve principalmente a relação "**encaminha paciente**" e "**atendimento**". Trata-se do caso em que uma **entidade de saúde** cuida de um **paciente** e precisa transferí-lo para outro **hospital**, o que é muito comum no mundo real, pois dependendo do caso, um certo **hospital** pode ter equipamentos, especializações ou mesmo certos médicos específicos para o paciente. Por conta disso, considera-se um ciclo natural, visto que o encaminhamento só pode ser feito a partir de uma entidade de saúde e deve ser para um hospital. Existe apenas uma exceção, que seria o caso de que uma entidade de saúde não pode encaminhar um paciente para si própria.

Ciclo: Gestor do Sistema → Requisição → Entidade de Saúde → Notificação

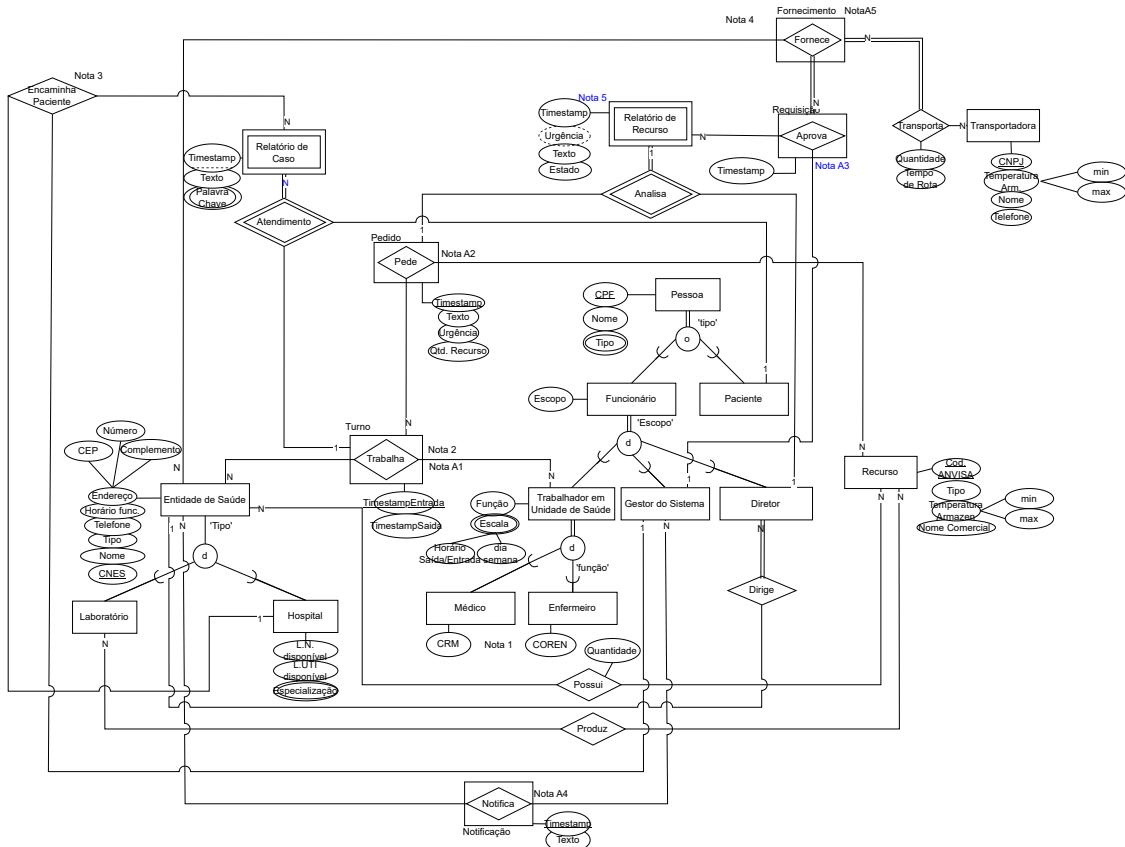
Ocorre por conta da dupla funcionalidade do **gestor** do sistema: ele pode **notificar** uma entidade de saúde quando várias entidades de saúde próximas a ela pediram por um mesmo recurso, podendo prever uma "**tendência**" na região. Ele também pode aprovar a **requisição** de recursos feita pela entidade de saúde onde ele trabalha. Vale destacar que as funcionalidades não são relacionadas, portanto trata-se de um ciclo natural da nossa modelagem.

Ciclo: Entidade de Saúde → Turno → Pedido → Relatório de Recurso → Requisição

Ocorre quando uma **entidade de saúde** gera uma requisição de recurso e outra entidade deve fornecer o recurso. Algo a ser tratado na aplicação é que uma mesma entidade de saúde não pode fornecer o recurso que ela mesmo pediu.

2.4 Diagrama do Modelo Entidade-Relacionamento

Para realização do projeto, foram usados como base os slides da Prof. Dra. Elaine Parros M. de Sousa, disponíveis na plataforma do Tidia-Ae 4.0, e a bibliografia da disciplina (ELMASRI; NAVATHE, 2011).



2.5 Mudanças relacionadas à Primeira Entrega

Após a análise da primeira entrega, algumas inconsistências foram identificadas pelo monitor. O grupo concordou com os apontamentos e realizou as devidas correções conforme descrito a seguir.

- O título da sessão 2.1, antes "Comportamento das entidades e dos relacionamentos", foi trocado para "Levantamento de Requisitos" e o texto dessa sessão foi levemente alterado para seguir um fluxo de leitura mais simples e direto. Foram tirados alguns detalhes confusos e irrelevantes, bem como algumas repetições de escrita que tornavam a leitura mais exaustiva e confusa. A ordem em que as entidades foram sendo apresentadas também foi alterado; tentou-se ao máximo não evocar entidades sem antes explicá-las.
- Foi-se ajustado a cardinalidade do relacionamento **Atendimento** em **Relatório de Caso**. Como o **timestamp** é chave da entidade fraca **Relatório de Caso**, a cardinalidade foi modificada para **1:1:N**, permitindo a geração de mais de um relatório de caso em um mesmo turno para um determinado paciente, com **timestamps** distintos.
- Em **Relatório de Recurso**, o atributo **timestamp** deixou de ser parte da chave primária, visto que um pedido pode gerar apenas um **relatório de recurso**. Assim, a unicidade é garantida pela **chave do pedido**, evitando problemas de consistência.
- Como um **relatório de recurso** pode gerar apenas uma **requisição**, a chave de **requisição** passou a ser composta exclusivamente pela chave do **Relatório de Recurso**. Os atributos **timestamp** e **Gestor** (que aprovou a requisição) foram **removidos da chave** para garantir consistência.

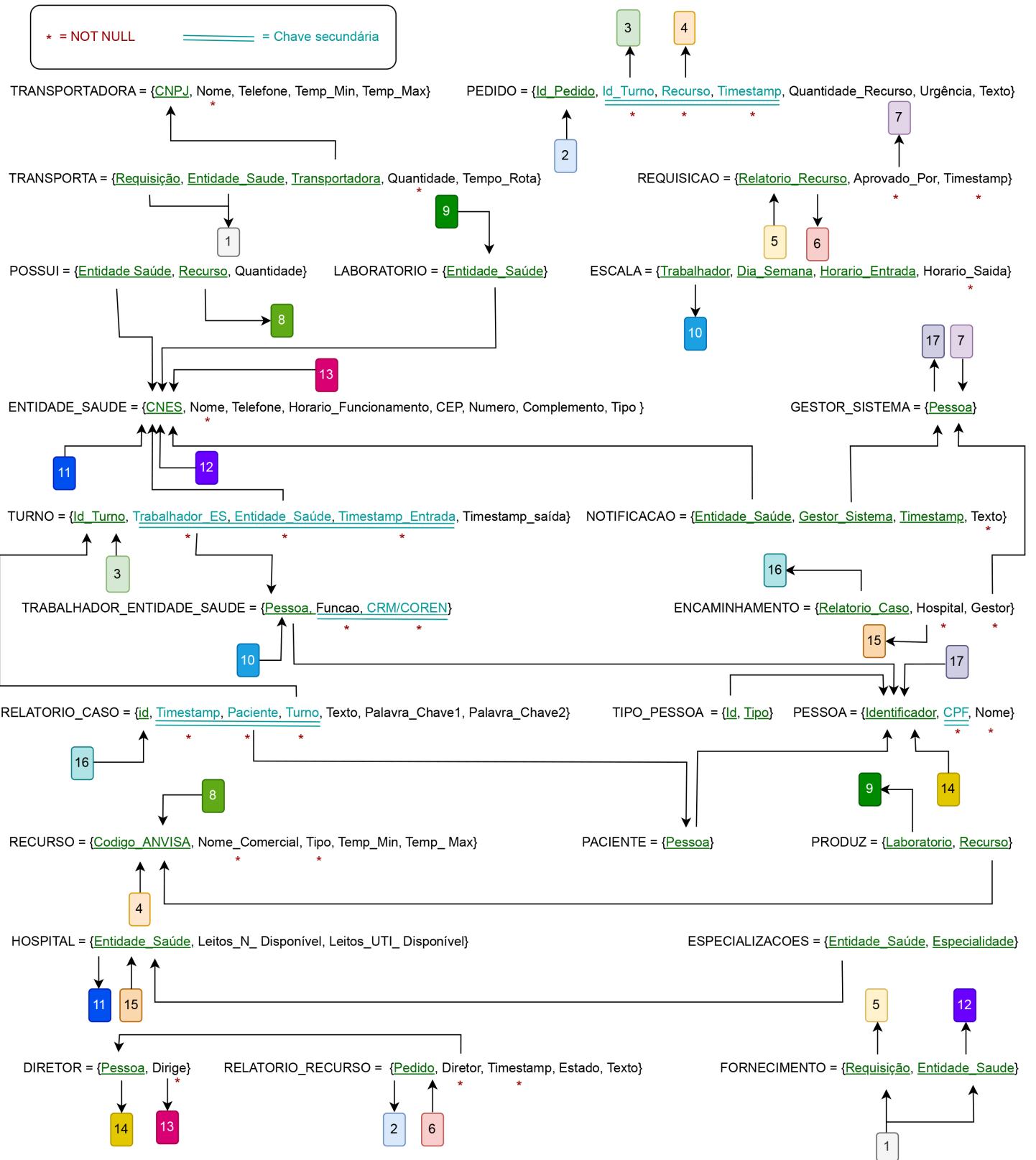
Além das inconsistências apontadas pelo monitor, o grupo julgou necessário realizar algumas outras alterações, as quais foram sendo adicionadas no decorrer do projeto. São elas:

- Foi acrescentado **participação total** de **diretor** na relação **dirige**, visando maior consistência dos dados. Apenas diretores ativos e que efetivamente dirigem uma entidade de saúde devem constar no banco.
- A modelagem de **períodos** e **dias da semana** foi ajustada, pois, durante o mapeamento, identificou-se um problema ao relacionar os atributos **períodos** e **dias da semana** da entidade **trabalhador em entidade de saúde**. Optou-se por substituí-los por um atributo **multivvalorado composto** chamado **escala**, permitindo associar diretamente o dia da semana e o horário de trabalho. Deve-se, entretanto, garantir que duas escalas não sejam sobrepostas.
- Foi incluído o atributo **Nome** em **Entidade de Saúde**, o qual foi adicionado por ser um atributo essencial para a visualização e identificação das **entidades de saúde** na aplicação.
- A relação **fornec** foi transformada em uma agregação (**fornecimento**) para assegurar a consistência do banco e permitir determinar exatamente qual **transportadora** está responsável por transportar o **recurso** de uma **entidade de saúde** em cada **requisição**. Segue um exemplo de inconsistência anterior: Transportadoras A e B podiam estar associadas à mesma requisição, que envolvia duas entidades de saúde fornecendo o recurso. Contudo, não havia como identificar qual transportadora estava vinculada a qual transporte. A agregação soluciona esse problema.

- O atributo Temperatura de Armazenamento foi decomposto em dois: **Temperatura Máxima** e **Temperatura Mínima**. Acreditamos que não fazia sentido guardar todas as informações de temperaturas possíveis que aquela transportadora consegue suportar, o mais adequado é guardar uma **faixa de valores** (variantes entre o valor máximo e mínimo). Assim o termos multivalorado foi substituído por um composto de dois atributos simples.

3 Modelo Relacional

Agora, será mostrado e explicado o Modelo Relacional obtido a partir do Modelo Entidade-Relacionamento pelo grupo desenvolvido, sendo que o primeiro relaciona as tabelas que serão geradas posteriormente. O modelo a seguir segue a notação usada em aula, sendo que as justificativas de escolhas de mapeamento seguem logo após o modelo:



3.1 Mapeamentos das Entidades e Generalizações

1. Generalização da entidade 'Pessoa'

- **Solução adotada:** O mapeamento da entidade **Pessoa** foi dado por meio de uma tabela separada para designação do seu tipo, uma tabela contendo os atributos gerais acompanhados de um identificador artificial (usado como chave primária substituta) e uma tabela para cada tipo específico de pessoa.
Por conta da forma como a entidade '**Funcionário**' foi mapeada, os valores do atributo **escopo** foram unidos aos de **tipo** de **Pessoa**. Dessa forma, os tipos mapeados na tabela **TIPO_PESSOA** passam a ser **Paciente**, **Trabalhador_ES**, **Diretor** e **Gestor**, correspondente aos CEE's homônimos.
- **Vantagens:** Essa forma de mapeamento, em conjunto com o identificador artificial (ID), garante maior segurança ao banco de dados por meio da centralização de informações sensíveis, como **Nome** e **CPF**, reduzindo o acesso e a replicação desses dados. Além disso, evita-se a redundância de atributos em casos de sobreposição.
A criação de uma tabela adicional com o tipo de **Pessoa** melhora o processo de busca pelo tipo (que agora também determina diretamente o escopo no caso de um funcionário) de uma pessoa.
- **Desvantagens:** Por conta da criação da tabela **Pessoa**, haverá uma aumento no tempo de busca em decorrência de uma junção adicional. Além disso, por consequência da introdução da tabela **TIPO_PESSOA**, torna-se necessário garantir a participação total das especializações na Entidade Geral e a disjunção entre os tipos correspondentes a um escopo de funcionário. Ressalta-se ainda que ocorre uma perda semântica na chave em virtude da adoção do identificador artificial (ID).
- **Alternativas:** Os conjuntos de entidades específicos (**CEE's**) poderiam ser mapeados diretamente, sem a criação de uma tabela **Pessoa**, o que reduziria o tempo de busca por exigir menos junções. No entanto, dados sensíveis, como o **CPF**, seriam replicados em diversas tabelas caso fossem utilizados como chaves. Também seria possível a omissão da tabela de tipos para economia de espaço em troca de um pior desempenho de busca pelo papel que a pessoa desempenha no banco.

2. Generalização da entidade 'Funcionário':

- **Solução adotada:** O Conjunto de Entidade Geral para este caso não possui atributos específicos e não participa de nenhum relacionamento, então, foi escolhido mapear apenas os Conjuntos de Entidades Específicos.
- **Vantagens:** Essa forma garante participação total e, como não há atributos gerais, não há desperdício de espaço. Além disso, o tempo de busca é otimizado, pois não há necessidade de junção com uma tabela geral.
- **Desvantagens:** É necessário garantir a disjunção entre as especializações no nível da aplicação, como explicitado no mapeamento de **Pessoa**, assegurando que um mesmo funcionário não tenha mais de um cargo.
- **Alternativas:** Poderia ser criada uma tabela para a entidade geral 'Funcionário', permitindo maior flexibilidade e padronização, mas isso geraria um aumento no tempo de busca devido às junções. Neste caso também não seria possível unir os atributos **escopo** e **tipo**.

3. Generalização de ‘Trabalhador em Entidade de Saúde’:

- **Solução adotada:** Como apenas o Conjunto Entidade Geral participa de relacionamentos e possui poucos atributos específicos, optou-se por mapear apenas o Conjunto Entidade Geral. Vale ressaltar que o atributo **COREM/CRM** foi condensado em uma única string. Além disso, os campos **Função** e **COREM/CRM** foram definidos como **NOT NULL**, sendo que atributo **função** foi considerado um booleano.
- **Vantagens:** Essa abordagem apresenta o melhor tempo de busca, pois todos os dados estão concentrados em uma única tabela. Como as especializações (**Médico** e **Enfermeiro**) possuem apenas um atributo específico cada (**CRM** e **COREM**, respectivamente), e ambos podem ser armazenados de forma semelhante, decidiu-se unificá-los em um único atributo, de forma a otimizar o espaço e minimizar valores nulos. Além disso, é garantida a disjunção e a participação total nessas especializações.
- **Desvantagens:** É de extrema importância garantir consistência na aplicação entre o tipo do trabalhador e o valor contido no campo **COREM/CRM**, evitando inconsistências semânticas.
- **Alternativas:** Poderia ser feita a criação de tabelas separadas para ‘Médico’ e ‘Enfermeiro’, o que facilitaria o controle semântico, porém aumentaria o custo de armazenamento e de consultas devido às junções.

4. Generalização de ‘Entidade de Saúde’:

- **Solução adotada:** Devido à participação não ser total, à grande quantidade de atributos gerais e específicos e ao fato de que tanto os conjuntos de entidades gerais (CEG) quanto os específicos (CEE) participam de relacionamentos, optou-se por mapear cada um dos CE em relações separadas.
- **Vantagens:** Essa estratégia facilita a modelagem dos relacionamentos de CEE e CEG e otimiza o uso de espaço, evitando o armazenamento de valores nulos.
- **Desvantagens:** É necessário garantir a disjunção entre os CE na aplicação. Além disso, o tempo de busca pelos CEE é maior, pois exige junção entre as relações.
- **Alternativas:** Poderia ser feita a união das entidades em uma única tabela com um atributo discriminador, reduzindo a necessidade de junções, mas com aumento de valores nulos e menor clareza semântica, além da dificuldade para representar as relações entre entidades.

3.2 Mapeamentos dos Relacionamentos

1. Relacionamento ‘Dirige’ com participação total de ‘Diretor’ - 1:N

- **Solução adotada:** Foi incluído um atributo **Dirige** dentro da tabela DIRETOR com valor **NOT NULL**. Este atributo é chave estrangeira da chave da relação ENTIDADE_SAUDE.
- **Vantagens:** Essa forma de mapeamento é a mais eficiente, pois elimina valores nulos, garante a participação total, mantém a consistência da cardinalidade e otimiza o tempo de busca, além de reduzir o consumo de espaço.
- **Desvantagens:** Não foram identificados desvantagens para este mapeamento.

- **Alternativas:** Poderia ter sido criada uma tabela separada para o relacionamento, porém haveria custo adicional de memória e replicação de dados, aumento no tempo de consulta, devido a junções adicionais, e perda da garantia de participação total.

2. Relacionamentos binários N:N

Nosso MER apresenta ao todo 3 relacionamentos binários N:N que não formam agregações: **Produz**, **Possui** e **Transporta**. Em todos esses casos, foi adotado a mesma solução de mapeamento.

- **Solução adotada:** Foi criada uma nova tabela para representar cada um desses relacionamentos, cuja chave primária é composta pelas chaves estrangeiras das entidades participantes do relacionamento. Além disso, as relações que apresentam atributos de relacionamento são mapeadas nessas tabelas.
- **Vantagens:** Esse mapeamento preserva a integridade e a flexibilidade para representar múltiplas associações entre as entidades.
- **Desvantagens:** Haverá aumento no tempo de busca, uma vez que consultas envolvendo essas relações requerem junções. Também vale dizer que, no caso da relação **Transporta**, não é possível garantir a participação total da entidade agregada **Fornecimento**
- **Alternativas:** Não encontramos outra alternativa válida que respeite todos os requisitos para este tipo de mapeamento

3. 'Atendimento' entre 'Paciente' e 'Turno' gerador de 'Relatório de Caso' (entidade Fraca) - 1:1:N

- **Solução adotada:** A relação, devido a presença da entidade fraca, foi mapeada inteiramente dentro da tabela da entidade **Relatório de Caso** (parte N da relação). Além disso, foi criado um identificador artificial (ID) dentro dessa tabela para diminuir o tamanho da chave primária, a qual é usada como chave estrangeira em outra relação.

A chave secundária são as chaves das entidades pertencentes a relação, com valor diferente de **NULL**, para garantir consistência no relacionamento.

- **Vantagens:** Facilidade no processo de busca devido a menor utilização de operações de junções e garantia da participação total de relatório de caso na relação.

A introdução do Identificador artificial economiza espaço na relação ENCAMINHA e melhora o tempo de busca.

- **Desvantagens:** Existe um problema proveniente da chave secundária relacionado ao **Turno**. Como é o trio (**Timestamp**, **Paciente**, **Turno**) que forma a chave secundária, ou seja, que é único, existe uma possibilidade que seja cadastrado ao mesmo tempo um paciente atendido em dois turnos diferentes, ou seja, que podem ter ocorrido em duas entidades de saúde diferentes, o que não faz sentido. Dessa forma, precisamos garantir consistência dessa relação em aplicação.

A introdução do Identificador artificial gera aumento de espaço dentro da própria relação.

- **Alternativas:** Poderia ser feita a criação de uma tabela intermediária para representar a relação ternária, porém não estariam levando em consideração que o relatório de casos é uma entidade fraca. Além disso aumentaríamos a complexidade e o custo de consultas.

4. 'Analisa' entre 'Pedido' e 'Diretor', gerador de 'Relatório de Recurso' (entidade fraca) - 1:1:1

- **Solução adotada:** O relacionamento **Analisa** foi mapeado inteiramente dentro da relação **Relatório de Recurso**, com a chave primária sendo a chave de pedido. As demais chaves estrangeiras provenientes da relação estão com o atributo **NOT NULL**.
- **Vantagens:** Tem-se garantia da participação total de **Relatório de Recurso** na relação, além de realizar uma busca mais eficiente.
- **Desvantagens:** Por **Relatório de Recurso** ser uma entidade fraca, é necessário controle dos registros em operações de exclusão ou atualização das entidades fortes.
- **Alternativas:** Poderia ter sido criado uma tabela adicional para representar a relação, o que tornaria o gasto em espaço maior e prejudicaria as operações de busca, com junção adicional.

5. 'Encaminha Paciente' entre 'Gestor', 'Hospital' e 'Relatório de Caso' - 1:1:N

- **Solução adotada:** Sabemos que nem todo **Relatório de Recurso** requer o encaminhamento de um paciente. Na verdade, são poucos os relatórios que exigem encaminhamento de fato. Por esse motivo, foi criada uma tabela adicional para representar a relação. Para garantir a consistência entre o relacionamento, a chave da relação é a chave de **Relatório de Caso**, já as demais chaves estrangeiras provenientes das outras entidades possuem a propriedade **NOT NULL**.
- **Vantagens:** Garantia de integridade da relação ternária e inexistência de valores nulos.
- **Desvantagens:** Devido a tabela extra, são necessários mais operações de junções para combinar os dados das tabelas em uma consulta, o que piora o tempo de busca.
- **Alternativas:** A relação poderia ter sido mapeada inteiramente dentro da entidade **Relatório de Caso**, o que melhoraria o processo de busca, mas perderia em espaço desperdiçado, com muitos valores nulos para os relatórios que não demandam encaminhamento, o que, conforme estipulado, são a maioria.

3.3 Mapeamentos das Agregações

1. Agregação 'Requisição' - 1:N identificada pela chave do CE de cardinalidade N

- **Solução adotada:** Foi criada uma nova relação, pois, embora a **Requisição** derive do relacionamento de **Aprovação** de um **Relatório de Recurso** por um **Gestor do Sistema**, foi considerado que muitos relatórios não são aprovados, o que levaria a **RELATORIO_RECURSO** ter muitos valores nulos, prejudicando a integridade e a eficiência do armazenamento.
- **Vantagens:** Essa abordagem reduz valores nulos e mantém a integridade dos dados, além de melhorar a rastreabilidade das aprovações, permitindo consultas mais diretas sobre as requisições emitidas.
- **Desvantagens:** O uso de uma relação adicional implica em maior número de junções em consultas que envolvem pedidos, relatórios e requisições, o que piora a busca. Além disso, é necessário controle a mais na aplicação para garantir que cada relatório gere no máximo uma requisição e que a coerência dos **Timestamps** sejam respeitadas.

- **Alternativas:** Uma alternativa seria armazenar os dados de aprovação diretamente em RELATORIO_RECUSO, adicionando os atributos **Aprovado_Por** e **Timestamp**, o que simplificaria as consultas e eliminaria a necessidade de junção, porém resultaria em uma grande quantidade de campos nulos - para relatórios não aprovados, além de dificultar o tratamento com o relacionamento **fornecce**.

2. Agregação ‘Fornecimento’ - N:N com participação total identificado pelas chaves dos CEs geradores

- **Solução adotada:** Como a agregação Fornecimento é identificada apenas pelas chaves dos CE geradores da agregação, foi criado uma tabela única para representar a relação **Fornece** e a agregação **Fornecimento**.
- **Vantagens:** A criação de uma tabela única melhora o tempo de busca, em relação a criação de duas tabelas, aumenta a flexibilidade do modelo e garante a integridade dos relacionamentos entre requisições, fornecimentos e transportes.
- **Desvantagens:** A participação total deve ser garantida em aplicação, o que pode dificultar o registro de estados intermediários, como requisições aprovadas ainda sem fornecedores definidos, exigindo maior controle lógico nesta etapa.
- **Alternativas:** Seria possível a criação de duas tabelas, uma para a relação **Fornece** e outra para a agregação **Fornecimento**, mas teríamos maior gasto de espaço e a busca seria prejudicada.

3. Agregação ‘Turno’ - N:N identificada por atributo próprio + chaves dos CEs do relacionamento gerador ‘Trabalha’

- **Solução adotada:** Como o relacionamento **Trabalha** não possui participação total nem atributos específicos, foi mapeada apenas a agregação **Turno**, utilizando uma chave composta pelas chaves estrangeiras das entidades geradoras mais o atributo próprio (**timestamp Entrada**). O campo **timestamp Saída** não faz parte da chave, pois no início do turno não se conhece o momento de término. Devido ao tamanho extenso da chave composta, foi criado um identificador artificial para otimizar espaço e desempenho nas consultas.
- **Vantagens:** Essa estratégia não gera redundâncias e permite representar de forma precisa os turnos ativos. Além disso, evita a criação de uma relação extra para ‘Trabalha’, economizando espaço.
A criação do identificador artificial reduz o tempo de busca e espaço gasto em outras relações que turno é chave estrangeira.
- **Desvantagens:** O volume elevado de registros na tabela ‘Turno’ torna as buscas por determinadas relações ‘Trabalha’ mais custosas. É também necessário garantir, a nível de aplicação, que um trabalhador não inicie um novo turno sem finalizar o anterior e que não ocorram sobreposições de períodos.
- **Alternativas:** Poderia ser criada uma relação separada para ‘Trabalha’, o que simplificaria algumas consultas, mas aumentaria o consumo de espaço e o tempo de busca por turnos.

Para a agregação **Pedido** - N:N identificada por atributo próprio + chaves dos CEs do relacionamento gerador **Pede** - foi adotada uma abordagem semelhante à de **Turno**. Da mesma forma, a agregação **Notificação** - N:N identificada por atributo próprio + chaves dos CEs do relacionamento gerador **Notifica** - seguiu a mesma lógica, porém com a diferença de que, neste caso, não há um identificador artificial. O motivo disso é que esta tabela não é referenciada por nenhuma outra.

3.4 Mapeamentos dos atributos

1. Atributo Multivalorado 'Escala' em 'Trabalhador em Entidade de Saúde'

- **Solução adotada:** Foi pensado que há uma alta variabilidade deste atributo, isto é, um **Trabalhador da Entidade de Saúde** pode trabalhar, por exemplo, nos 7 dias da semana e em mais de uma **Entidade de Saúde** por dia (manhã, tarde e noite), o que resultaria em 21 instâncias diferentes para o mesmo **Trabalhador**. Da mesma forma, o **Trabalhador** pode trabalhar em apenas uma **Entidade** das cadastradas no banco e por 1 só dia, o que resultaria em apenas uma instância. Por esse motivo, optou-se por mapear este atributo em uma relação separada.
Além disso, foi escolhido não deixar o atributo de **Horario Saida** como chave da relação, o que evita, por exemplo, inconsistência de um **Trabalhador** estar cadastrado para trabalhar na quinta-feira das 14h às 18h e também das 14h às 16h, o que não faz sentido. A ideia é que a definição do dia da semana e do horário de entrada seja suficiente para definir o horário de saída (não há um "grau de liberdade" para este atributo), o qual pode ser nulo também.
- **Vantagens:** Essa forma oferece maior versatilidade na hora do cadastro da **Escala** do **Trabalhador**, evitando o armazenamento de valores **NULL** no banco. Assim, o modelo consegue se adaptar melhor a diferentes cargas de trabalho.
- **Desvantagens:** Da forma como está modelada, o tempo de busca se torna mais custoso, pois é necessário realizar junções entre as tabelas. Também, não é possível garantir que **Escalas** inconsistentes não sejam criadas - com o horário de saída sendo anterior ao horário de chegada, ou que haja sobreposição dos horários. Estes problemas devem ser tratados em aplicação.
- **Alternativas:** Poderia ter sido utilizado um número fixo de atributos dentro da tabela **TRABALHADOR_ENTIDADE_SAÚDE**, o que poderia facilitar as buscas, mas aumentaria substancialmente a quantidade de valores nulos e/ou reduziria drasticamente a flexibilidade das cargas de trabalho.

2. Atributo Multivalorado 'Especialidades' na entidade 'Hospital'

- **Solução adotada:** A justificativa para esta modelagem segue um padrão parecido ao adotado em **Escala**. Dado que existem muitos hospitais especializados e outros mais generalistas, optou-se por mapear este atributo em uma relação separada.
- **Vantagens:** Como antes dito, essa modelagem oferece maior versatilidade e evita o armazenamento de valores nulos.
- **Desvantagens:** O tempo de busca se torna mais custoso, pois é necessário realizar junções entre as tabelas.
- **Alternativas:** Poderia ter sido utilizado um número fixo de atributos na mesma tabela para representar as especialidades, mas isso poderia aumentar a quantidade de valores nulos e reduzir a flexibilidade.

3. Atributo Multivalorado 'Palavra-Chave' em 'Relatório de Caso'

- **Solução adotada:** Seria agradável que os profissionais que geram os **Relatórios de Casos** fossem mais precisos na definição de palavras-chave que os definem; atrelando apenas informações como o nome da doença que o paciente apresenta ou algum sintoma característico. Isso facilitaria uma possível análise futura do **Gestor do Sistema**, visto que o processo de reconhecimento de tendências epidemiológicas é

realizado por buscas frequentes e por todo o banco pelos **Relatórios de Casos**. Dessa forma, o número de palavras-chave atrelado ao relatório foi restringido para apenas 2, como atributos dentro da relação RELATORIO_CASO.

- **Vantagens:** O processo de busca pelas palavras-chave é melhorado, evitando que sejam necessárias junções para realização de consultas pelas palavras-chave.
- **Desvantagens:** Existe a possibilidade de muitos valores nulos na tabela, o que aumenta a quantidade de espaço em memória desperdiçado. Da mesma forma, a versatilidade do profissional em registrar mais que duas palavras-chave é restringida, caso isso seja importante.
- **Alternativas:** Uma tabela extra para registrar as palavras-chave do relatório de casos poderia ser criada, o que melhoraria a flexibilidade deste atributo e diminuiria o espaço gasto em valores nulos, mas pioraria o processo de busca.

4. Atributo Derivado ‘Urgência’ em ‘Relatório de Recurso’:

- **Solução adotada:** Foi escolhido não armazenar fisicamente no banco de dados o atributo, calculando sob demanda, uma vez que o valor depende de outras informações já persistidas e pode ser obtido dinamicamente por meio de consultas ou procedimentos.
- **Vantagens:** Estamos reduzindo o espaço de armazenamento.
- **Desvantagens:** O cálculo sob demanda pode aumentar o custo computacional em consultas.
- **Alternativas:** Uma alternativa seria armazenar o valor de **Urgência** fisicamente na tabela de **Relatório de recurso**, o que ganharia em consulta, mas perderia em espaço.

3.5 Mudanças relacionadas à Segunda Entrega

Após a análise da segunda entrega e durante a confecção da terceira, optamos por algumas modificações no mapeamento relacional as quais serão detalhadas a seguir:

- Na relação **TRABALHADOR_ENTIDADE_SAUDA**, o atributo **Função** foi adicionado à chave secundária junto da imposição de valor **NOT NULL**. Essa mudança foi motivada pelo fato de que o valor de um **COREM** e de um **CRM** podem ser iguais para funções diferentes. Isso é, há a possibilidade de um médico com um **CRM** idêntico ao **COREM** de um enfermeiro.
- Além disso, adicionamos a condição de **NOT NULL** a alguns atributos. A decisão se deu em razão da perda semântica e/ou lógica da inserção de registros que não apresentem esses valores definidos.

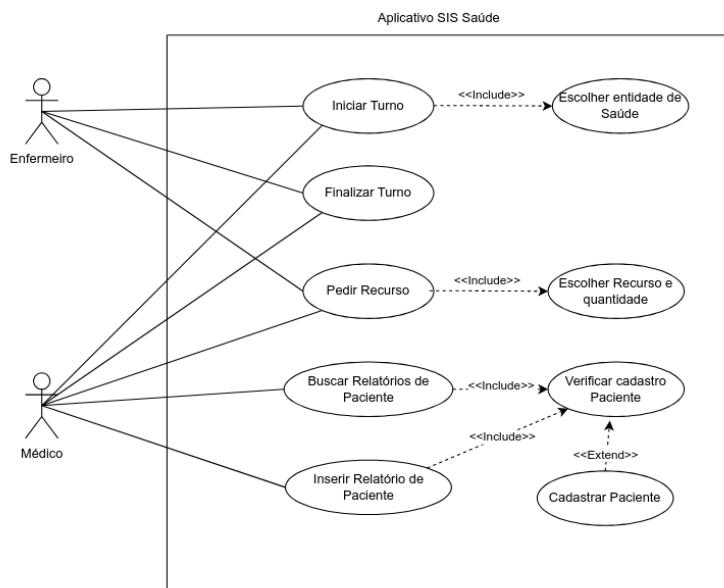
Estes foram:

- **Nome** em **PESSOA**
- **Nome** em **ENTIDADE_SAUDA**
- **Nome_Comercial** e **Tipo** em **RECURSO**
- **Horario_Saida** em **ESCALA**
- **Texto** em **NOTIFICACAO**
- **Nome** em **TRANSPORTADORA**
- **Quantidade** em **TRANSPORTA**

4 Implementação

4.1 Aplicação

Para a aplicação decidimos implementar um sistema que deve ficar disponível dentro de um hospital para que enfermeiros e médicos sejam capazes de interagir com o banco integrado de todas as unidades de saúde presente no Sis-Saude. Abaixo pode se observar um diagrama de caso de uso da aplicação:



Requisitos do sistema

Utilizou-se o sistema gerenciador de banco de dados **PostgreSQL** na versão **17.6**. Já para a aplicação, utilizamos **Python** juntamente com a biblioteca **Psycopg 3**, responsável pela comunicação da aplicação com o banco de dados. Essa biblioteca permite realizar queries de forma segura contra ataques de **SQL Injection**, pois utiliza parametrização de comandos, onde valores enviados pelo usuário não são concatenados diretamente na string SQL, mas sim passados como parâmetros isolados. Segundo a documentação oficial, o Psycopg 3 substitui automaticamente esses parâmetros de forma segura usando **placeholders** (%s), garantindo que o valor nunca seja interpretado como parte do comando SQL — apenas como dado. Assim, mesmo que o usuário tente inserir conteúdo malicioso, o banco trata tudo como valor literal, preservando a integridade da query.

Outro aspecto importante da aplicação é o cuidado ao inserir dados que dependem de múltiplas tabelas. Por exemplo, ao cadastrar um novo relatório de caso para um paciente novo, é necessário inserir primeiro na tabela **PESSOA**, depois em **TIPO_PESSOA**, depois em **PACIENTE**, e apenas então no **RELATORIO_DE_CASO**. Se alguma dessas etapas falhar, as inserções anteriores já teriam sido gravadas, criando inconsistências no banco.

Para evitar esse problema, utilizamos o método **transaction** da classe **DatabaseConnection**, que encapsula todas essas operações em uma única transação atômica. Dentro do bloco **with self.transaction():**, o Psycopg só executa o commit se todas as inserções forem bem-sucedidas; caso ocorra qualquer erro, é feito um rollback automático, desfazendo tudo e garantindo que o banco permaneça consistente.

Instruções para uso

No repositório do GitHub pode-se encontrar um tutorial, no **README**, de como rodar localmente o programa: Depois da criação do banco de dados e inserção dos dados artificiais presentes no repositório, é possível utilizar dados da tabela **TRABALHADOR_ES** para encontrar o CRM de um médico ou enfermeiro para prosseguir com a aplicação. Desde a criação para finalização de um turno até o pedido de um recurso ou geração de relatório de caso de um paciente.

4.2 Script de consultas

C1. Para todos os recursos, consultar os hospitais que possuem estoque abaixo da média para cada recurso

```
SELECT
    h.cnes_hospital, r регистрація_ms, r.name,
    COALESCE(p.quantity_disponible, 0) AS qtd_hospital,
    media.avg_stock, media.maior_stock, media.menor_stock
FROM hospital h CROSS JOIN recurso r
LEFT JOIN possui p ON (p.cnes_entidade_saude = h.cnes_hospital
    AND p регистрація_ms_recurso = r регистрація_ms)
JOIN (
    SELECT
        r2 регистрація_ms, AVG(COALESCE(p2.quantity_disponible, 0)) as avg_stock,
        MIN(COALESCE(p2.quantity_disponible, 0)) as menor_stock,
        MAX(COALESCE(p2.quantity_disponible, 0)) as maior_stock
    FROM hospital h2
    JOIN possui p2 ON h2.cnes_hospital = p2.cnes_entidade_saude
    RIGHT JOIN recurso r2 ON r2 регистрація_ms = p2 регистрація_ms_recurso
    GROUP BY (r2 регистрація_ms)
) media ON media регистрація_ms = r регистрація_ms
WHERE (COALESCE(p.quantity_disponible, 0) < media.avg_stock
    OR COALESCE(p.quantity_disponible, 0) = 0)
ORDER BY h.cnes_hospital;
```

Para realizar esta consulta, inicialmente cruzamos todas as combinações possíveis entre hospitais e recursos, utilizando o comando **CROSS JOIN**. Essa operação garante que todos os recursos sejam associados a todos os hospitais, mesmo que determinado recurso não esteja presente em seu estoque.

Em seguida, utilizamos um **LEFT JOIN** com a relação **POSSUI**, responsável por informar a quantidade disponível de cada recurso em cada hospital. Optamos por **LEFT JOIN** para preservar as tuplas em que o recurso não está registrado no estoque. Para tratar valores ausentes, aplicamos a função **COALESCE**, substituindo **NULL** no atributo **quantity_disponible** por 0.

O resultado dessas junções é então combinado, por meio do identificador do recurso, com os dados retornados por uma consulta interna **não correlacionada**. Essa subconsulta calcula, para cada recurso cadastrado no banco de dados (independente de sua presença nos estoques dos hospitais), os seus valores **mínimo**, **máximo** e **médio** disponíveis.

Por fim, aplicamos a cláusula **WHERE** para selecionar apenas os recursos que estão com estoque abaixo da média ou com quantidade igual a zero. Os resultados são retornados de forma ordenada pelo código **CNES** do hospital. A saída da consulta apresenta uma tupla contendo:

- o identificador do hospital,

- o código e o nome do recurso com baixa disponibilidade,
- sua quantidade em estoque, e
- os dados estatísticos gerais do recurso (média, valor mínimo e valor máximo em estoque nos hospitais).

C2. Pesquisar pelos pedidos sob análise que requisitarem por uma quantia de um determinado recurso acima da disponível no estoque total

```

SELECT
P.id_pedido, R.nome, R регистрация_ms, P.quantidade,
((COALESCE(EXT.estoque_total, 0) - COALESCE(PS.quantidade_disponivel, 0)) AS estoque_externo,
((COALESCE(EXT.estoque_total, 0) - COALESCE(PS.quantidade_disponivel, 0)) - P.quantidade) AS em_falta
FROM relatorio_recurso RR
JOIN pedido P ON P.id_pedido = RR.id_pedido_relatorio
JOIN turno T ON P.id_turno = T.id_turno
JOIN recurso R ON R регистрация_ms = P регистрация_ms_recurso
LEFT JOIN possui PS ON PS.cnes_entidade_saude = T.cnes_entidade_saude
    AND PS регистрация_ms_recurso = P регистрация_ms_recurso
LEFT JOIN (
    SELECT SUM(P0.quantidade_disponivel) AS estoque_total, P0 регистрация_ms_recurso AS recurso
    FROM possui P0
    GROUP BY P0 регистрация_ms_recurso
) AS EXT ON EXT.recurso = P регистрация_ms_recurso
WHERE P.quantidade > ((COALESCE(EXT.estoque_total, 0) - COALESCE(PS.quantidade_disponivel, 0)))
    AND RR.estado_relatorio = 'ANALISE'

```

Para realizar esta consulta, primeiramente é necessário obter o **estoque total de cada recurso no banco**, informação calculada por meio de uma **subconsulta não correlacionada**. Destacamos que os valores nulos são tratados utilizando a função **COALESCE**, substituindo-os por **0**, de forma a evitar inconsistências nos cálculos.

Na consulta principal, utilizamos diversos **JOINS** para relacionar os relatórios de recursos com o **estoque do hospital responsável pelo pedido**, informação armazenada na tabela **POSSUI**.

Com esses dados, conseguimos identificar a **quantidade do recurso solicitada** pelo hospital; e a **quantidade total disponível em estoque nos demais hospitais**, desconsiderando o hospital que fez a requisição.

Assim, comparamos o estoque externo do recurso com a quantidade solicitada, a fim de verificar se a demanda **excede a disponibilidade**. Caso isso ocorra, o recurso é retornado como resultado da consulta.

C3. Selecionar os recursos que são produzidos por laboratórios mas nenhum hospital tem em estoque

```
SELECT R регистрация_ms, R.nome, R.tipo
FROM recurso R
JOIN produz P ON
    R регистрация_ms = P регистрация_ms_ресурса
LEFT JOIN (
    SELECT DISTINCT P1 регистрация_ms_ресурса
    FROM possui P1
    JOIN entidade_saude ES
        ON P1.cnes_entidade_saude = ES.cnes
    WHERE ES.tipo_entidade = 'HOSPITAL'
        AND P1.quantidade_disponivel > 0
) AS RecursosEstoque
ON R регистрация_ms = RecursosEstoque регистрация_ms_ресурса
WHERE RecursosEstoque регистрация_ms_ресурса IS NULL;
```

Para realizar esta consulta, seguimos três etapas lógicas principais. Inicialmente, pegamos o conjunto de todos os recursos que são fabricados por pelo menos um laboratório, o que é realizado através do **JOIN** entre a tabela **RECURSO** e a tabela **PRODUZ**. Este **JOIN** é necessário para recuperar as informações do recurso.

Em seguida, empregamos um **LEFT JOIN** com uma subconsulta (não correlacionada), que serve para identificar os recursos com estoque em hospitais, os quais, **não deverão ser incluídos** no resultado. Assim, o **LEFT JOIN** tenta correlacionar cada recurso produzido por laboratórios com a lista de recursos em estoque nos hospitais. Se um recurso não for encontrado na subconsulta, o **LEFT JOIN** gera valores **NULL** para as colunas da subconsulta.

Por fim, a cláusula **WHERE** utiliza o resultado do **LEFT JOIN** para selecionar apenas os recursos que resultaram em valores **NULL**: garantindo que o recurso é produzido por um laboratório e não está na lista de recursos com estoque positivo em nenhum hospital.

C4. Selecionar todos os hospitais que apresentam as 3 especialidades menos frequentes

```
SELECT H.*  
  FROM HOSPITAL H  
 WHERE NOT EXISTS (  
   (SELECT E1.ESPECIALIDADE  
    FROM ESPECIALIZACOES E1  
    GROUP BY E1.ESPECIALIDADE  
    ORDER BY COUNT(*) ASC  
    LIMIT 3)  
 EXCEPT  
   (SELECT E.ESPECIALIDADE  
    FROM ESPECIALIZACOES E  
    WHERE E.cnes_hospital = H.cnes_hospital)  
 ) ;
```

Esta consulta realiza uma operação de **divisão relacional**, assim como a consulta seguinte. Para demonstrar diferentes formas de resolver esse tipo de operação, utilizamos duas abordagens distintas. Nesta primeira abordagem, aplicamos o **método de subtração de conjuntos**.

Primeiro, identificamos as **3 especialidades menos frequentes** no banco de dados. Para isso, agrupamos os registros pela especialidade dos hospitais e contamos o número de ocorrências de cada uma. Em seguida, ordenamos o resultado de forma crescente e aplicamos **LIMIT 3** para obter apenas as três especialidades menos comuns.

Na sequência, realizamos uma **consulta correlacionada**, a qual identifica quais especialidades estão associadas a um determinado hospital.

Por fim, aplicamos a operação de **subtração de conjuntos** utilizando o operador **EXCEPT**. Dessa forma, se o hospital analisado possuir **todas** as especialidades menos frequentes (com compatibilidade de domínio), então nada restará após o EXCEPT e a condição **NOT EXISTS** será satisfeita, resultando na seleção desse hospital.

C5. Consultar os laboratórios capazes de produzir todos os recursos requisitados por um dado hospital

```
WITH recursos_requisitados AS (
    SELECT DISTINCT p регистрация_ms_ресурс
        FROM отчет_ресурс r
        JOIN заказ p ON r.id_pedido_relatorio = p.id_pedido
        JOIN смена t ON p.id_смена = t.id_смена
    WHERE (t.cnnes_entidade_saude = '2751503'
        AND r.результат_отчета = 'ANALISE'))
SELECT cnes_laboratorio, COUNT(*)
    FROM производит
WHERE регистрация_ms_ресурс IN
    (SELECT * FROM recursos_requisitados)
GROUP BY cnes_laboratorio
HAVING COUNT(*) = (SELECT COUNT(*) FROM recursos_requisitados);
```

Esta consulta também realiza uma **divisão relacional**, porém utiliza uma abordagem diferente da consulta anterior — neste caso, baseada **na contagem de tuplas**.

Primeiramente, definimos uma subconsulta independente chamada **recursos_requisitados**, responsável por identificar todos os recursos distintos que estão sendo analisados para um determinado hospital. O uso da cláusula **DISTINCT** garante que cada recurso seja considerado apenas uma vez.

Na consulta principal, acessamos a tabela **PRODUZ** e aplicamos a cláusula **WHERE** com o operador **IN**, filtrando apenas os laboratórios que pelo menos um dos recursos identificados na subconsulta.

Em seguida, agrupamos os resultados por laboratório e utilizamos **COUNT(*)** para contabilizar quantos dos recursos requisitados cada laboratório é capaz de produzir.

Por fim, selecionamos apenas os laboratórios cuja quantidade de recursos produzidos é **exatamente igual** ao número total de recursos distintos retornados pela subconsulta.

5 Conclusão

Em geral, acreditamos que conseguimos atender a todos os requisitos pedidos pelo trabalho, e, portanto, consideramos que obtivemos êxito. Nossa experiência foi muito positiva e enriquecedora, sendo que todos os membros participaram ativamente das aulas, dos atendimentos, das discussões de planejamento do grupo e da aplicação prática dos conceitos estudados ao longo da disciplina.

Falando um pouco sobre o andar da disciplina, o grupo como um todo considera que as aulas foram excelentes, sendo que a professora conseguiu explicitar e ilustrar muito bem os conteúdos para as provas e para o desenvolvimento do trabalho, sendo que este abrangeu a maioria dos temas estudados em sala, o que nos ajudou a entender a aplicação real da disciplina como um todo.

Os atendimentos foram muito esclarecedores e assertivos em relação as dúvidas que surgiram durante o desenvolvimento, sendo essenciais para a entrega.

Acreditamos que as provas foram relativamente extensas, porém abordaram todo o conteúdo dado em aula. O trabalho também conseguiu abordar a maioria do conteúdo. Um

ponto negativo que o grupo levantou foi a carga puxada da disciplina. As provas e o trabalho individualmente foram coerentes, porém, ao somar os dois, sentimos que ficou bem pesado.

Apesar de tudo, gostaríamos de agradecer a professora Elaine e ao monitor André por todo o conhecimento e tempo disponibilizado para nós, de forma que notamos uma grande evolução em relação ao começo da disciplina.

Referências

ABDALA, Vitor. **Perda de insumos do Ministério da Saúde soma R\$ 2 bilhões desde 2019.** Agência Brasil. Abr. 2023. Disponível em:
<https://agenciabrasil.ebc.com.br/saude/noticia/2023-04/perda-de-insumos-do-ministerio-da-saude-soma-r-2-bilhoes-desde-2019>. Acesso em: 5 set. 2025.

ANJOS, Adalton dos. **Novo índice aponta que desigualdades sociais em saúde no Brasil se aprofundaram na pandemia.** Fiocruz - CIDACS. Jun. 2022. Disponível em:
<https://cidacs.bahia.fiocruz.br/2022/06/novo-indice-aponta-que-desigualdades-sociais-em-saude-no-brasil-se-aprofundaram-na-pandemia/>. Acesso em: 5 set. 2025.

ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistemas de banco de dados.** 6. ed. [S.l.]: Addison-Wesley, 2011.