

Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação

Base unificada PyLadies Brasil
Descrição do Problema e Modelagem

SSC0640 - Bases de Dados

Prof.^a Dr.^a Elaine Parros Machado de Souza

Luiz Fernando Silva Eugênio dos Santos 10892680

Samuel Rubens Souza Oliveira 11912533

São Carlos, Agosto de 2022

1 Introdução

A área de tecnologia da informação é conhecida pela disparidade da atuação entre mulheres e homens na área, em 2020 foi constatado pelo Instituto AnitaB.org que mulheres compõem apenas 28,8% da força de trabalho em TI. Apesar dessa proporção estar crescendo gradualmente, a representatividade feminina ainda é pequena, o que acaba desestimulando mulheres a entrarem nesta área.

Neste contexto diversas iniciativas foram criadas para incentivar mulheres a aprenderem programação e adentrarem nesta carreira. Como exemplos, temos o *Django Girls*, o *she++*, o *WiT (Women In Tech)*, e o *PyLadies* que será o foco deste projeto.

O PyLadies é uma organização internacional sem fins lucrativos, com foco em incentivar mais mulheres a se tornarem participantes ativas e líderes na comunidade tecnológica. A missão da ONG é promover, educar e impulsionar a existência de uma comunidade Python diversificada através de sensibilização, educação, conferências, eventos e encontros sociais. O objetivo não é segregar, mas ser um meio de integrar as mulheres à programação.

O PyLadies se organiza em **Capítulos** (tradução literal de chapters), ou seja cada capítulo é uma instância do grupo em uma determinada localização (como cidade, região ou país). Para criar um Capítulo é necessário requisitar o pedido para a organização PyLadies Global, o pedido é aceito quando não há capítulos com o mesmo nome, sendo necessário duas pyladies representantes no momento da criação. Depois de criado, qualquer mulher ou pessoa não binária pode se juntar ao capítulo se tornando uma *PyLady*.

Um capítulo não possui quantidade máxima de integrantes, as pyladies representantes também são participantes do grupo. A *PyLady* que é organizadora não possui vantagens em relação a outras participantes no quesito decisões do grupo, no entanto ela é responsável por organizar ou liderar atividades, podendo ser consultada para esclarecer dúvidas sobre estas.

Os capítulos realizam atividades, podendo estas serem para o público interno do capítulo (Projetos Internos). externo (Eventos, qualquer pessoa pode participar desde se inscreva) e até mesmo para seus colaboradores (projetos colaborativos). Para os projetos internos não é necessário controle de inscrições, podendo inclusive ser informal (Exemplo: tutorial de GitHub ministrado por uma *pylady* para ensinar as outras PyLadies como utilizar a ferramenta em prol do grupo). Já os eventos, por serem abertos, é necessário o cadastro como participante (mesmo que seja uma *pylady*). Os eventos podem ser Palestras, Workshops, Minicursos e qualquer outro tipo de atividade que seja fornecido à comunidade externa pelo capítulo.

Além disso os capítulos podem se relacionar com seus colaboradores, que são empresas e outras entidades. Esse relacionamento se trata de uma contribuição entre ambas as partes que pode ser do tipo patrocínio ou uma troca de atividades, tendo sempre as condições da contribuição documentadas em um acordo. Quando se trata do oferecimento

de um patrocínio, este costuma ser de forma monetária, mas pode ocorrer de outras maneiras como fornecimento de materiais físicos, ingressos para eventos, materiais de estudo e etc.

Já a colaboração, que não é um patrocínio, é uma troca de atividades acordada entre ambas as partes, desta forma o capítulo oferece uma atividade para o colaborador e o mesmo oferece uma atividade para o grupo, sendo estes chamados, Projetos Colaborativos, que podem vir a se tornar também um Projeto Interno ou um Evento.

A comunidade PyLadies também possui um conselho global (PyLadies Global Council) responsável por intervir em um capítulo em casos extremos, em que seja ferido o código de conduta da comunidade ou que infrinja as leis de seu país. Ademais o PyLadies Global Council também pode gerir seus próprios projetos.

Portanto, este projeto visa modelar uma base de dados que represente a comunidade brasileira de PyLadies, evitando dessa maneira inconsistência de dados entre diferentes países. Vale ressaltar que a comunidade PyLadies atualmente não possui uma base de dados que integra os capítulos e suas participantes. Portanto este projeto também poderá contribuir para a criação de uma base de dados para a comunidade brasileira de *PyLadies* e servir como inspiração para a comunidade PyLadies Global.

2 Modelagem

2.1 Requisitos e funcionalidades

Para delimitarmos as funcionalidades, começamos considerando a entidade **Capítulo** isoladamente, com seus componentes, ações e necessidades. Distinguimos os capítulos por seu nome e sua localização, de forma que estes são os atributos usados para integrar a chave composta de capítulo. Os capítulos também possuem como atributo o número de integrantes (que varia com o tempo), as frentes que o compõe (tesouraria, marketing, entre outros) podendo variar de um capítulo para o outro e o website que o representa. Os capítulos realizam diversas atividades sejam elas para o público externo, interno ou para os colaboradores e essas atividades podem ser realizadas por um único capítulo ou por uma colaboração entre diferentes capítulos (ex.: pyladies São Carlos e Pyladies São Paulo realizam um evento juntos).

A entidade **Pessoa** tem como identificador o atributo E-mail, pois dados pessoais como CPF, RG, entre outros não são obrigatórios para cadastro na Base de Dados do Pyladies. A entidade pessoa pode ser do tipo pylady e participante (Generalização com sobreposição), podendo inclusive uma pessoa se cadastrar sem ser uma das opções acima. A entidade possui também os atributos Nome, Idade, Gênero, Etnia, Cor, Ocupação, CPF, Orientação Sexual e Tipo.

A entidade **Pylady**, pode representar e participar de no máximo um capítulo. Além disso, a pylady pode organizar várias atividades realizadas pelo capítulo (no seu atributo Atividades coordenadas é possível verificar todas as atividades que a pylady já organizou), uma pylady pode ser participante de eventos e pode apresentar varias modalidades de evento (pode ser palestrante, ministradora e monitora de vários eventos). Há também um atributo que guarda a faixa salarial que cada pylady.

A entidade **Participante** esta ligada a entidade **Evento** (herança de Atividades) por meio do relacionamento inscreve-se, ou seja, só pode participar de determinado evento se for um participante e estar devidamente inscrito no evento em questão, essa relação sempre irá gerar um certificado para o participante. O participante pode se inscrever em mais de um evento, é possível verificar quais os eventos que o participante está inscrito em seu atributo Eventos inscritos.

A entidade **Atividade** tem como owner a entidade **Capítulo** e possui como chaves parciais a composição dos atributos Título e Data (é necessário saber qual capítulo esta realizando uma atividade com determinado titulo e data). Há os atributos local (onde ocorre a atividade), descrição, duração ,tipo , responsável (que é uma pylady) e os participantes (número de participantes numa atividade). As atividades realizadas pelos capítulos só podem ser do tipo Eventos, projeto interno e projeto colaborativo .

A entidade **Evento** é um tipo de atividade com público livre (qualquer pessoa que se cadastre como participante está apto a se inscrever). Pode ser classificado como Palestra (modalidade palestra) que possui um tema e o nome de seus palestrantes, Workshop (modalidade workshop) que possui atributos para guardar os materiais necessários e o nome de seus ministrantes e Curso(modalidade curso) que possui atributos para guardar o nome de seus ministrantes e monitores. Cabe salientar, que apenas pyladies podem ser palestrantes, ministrantes ou monitoras (esse controle vem da relação **Apresenta** entre a entidade **Evento** e **Pylady**, uma pylady pode apresentar mais de uma modalidade de evento) e que estes atributos, juntamente com Material necessário, estão sendo tratados como uma 'String Gigante', visto que essas informações não são tão relevantes para a base de dados .

A entidade **Projeto Interno** são atividades para comunidade interna do Pyladies (em especial para um único capítulo) são feitos com o intuito de passar conhecimento entre as pyladies ,podendo inclusive serem informais. O link relatório será onde cada pyladie submeteria seu trabalho no final de cada projeto.

A entidade **Projeto colaborativo** são atividades de "troca" entre um capítulo pyladies e seus colaboradores (empresas, ONG's , grupos de extensão, entre outros). Um capítulo oferece determinado minicurso, workshop ,palestra, etc.. para um colaborador e ele dá em troca um minicurso, palestra, etc... para o capítulo. Esses retornos dos colaboradores podem ser utilizados para projetos internos ou eventos do capítulo (as pyladies que delegam

como será usado). Há o atributo **Colaboradores** que guarda o nome dos colaboradores que estão em determinado projeto colaborativo.

Em sequência, iremos dar foco as entidades **Colaboradores** e **Colaboração**. Os colaboradores colaboram com diversos capítulos (quantas vezes quiser), seja com recursos financeiros, materiais tecnológicos/estudos, ingressos para eventos ,entre outros.

A entidade **Colaboradores** (herança com disjunção) tem como chave composta seu nome e sua cidade, além de atributos para contato como e-mail e telefone. Seu tipo pode ser Empresa que possui um CNPJ, grupo de extensão que possui uma universidade atrelada e um objetivo (o que o grupo faz) ou ONG que possui o atributo **Objetivo** com semântica semelhante ao de grupo de extensão.

Cada par da relação colabora (entre as entidades **Capítulo** e **Colaboradores**) gera uma colaboração diferente que é identificada por sua datatime. Mediante isso, a entidade agregada **Colaboração** tem como intuito identificar cada uma das colaborações realizadas (ex.: um mesmo colaborador pode doar para um mesmo capítulo quantas vezes quiser,o que diferencia cada colaboração é a data e a hora). Essa colaboração possui um acordo que define se o colaborador quer ou não uma divulgação em troca, caso seja exigida uma divulgação o atributo booleano **isPatrocínio** se torna verdadeiro, visto que patrocínio nada mais é que uma colaboração que exige que o capítulo faça uma divulgação do colaborador em questão. Uma atividades pode resultar de várias ou nenhuma colaboração.

A entidade **Doação** pode ser anonima ou não, devido a essa possibilidade ela tem como owner a entidade capítulo e tem como chaves parciais a composição de Data e Hora, isso nos garante saber a qual capítulo determinada doação esta se destinando. Caso ela não seja anonima , podemos atrelar ela a entidade **Pessoa**, é válido lembrar que podemos cadastrar qualquer pessoa que não seja pylady ou participante.

A entidade **PyLadies Global Council** refere-se ao grupo de 0 a 3 PyLadies brasileiras que podem ser eleitas ao cargo, esta entidade possui como chave **ano** pois a cada 2 anos é criado um novo conselho através de eleições. Desta forma define-se a relação **Pertence** entre **PyLady** e **PyLadies Global Council** que se torna também a agregação **Mandato** pois a mesma **PyLady** pode pertencer a todos os **PyLadies Global Council** em que ela for eleita. Além disso o **PyLadies Global Council** pode tomar ações sobre qualquer **Capítulo** quando necessário, sendo esta relação intitulada **Delibera**.

2.2 Principais operações

De acordo com os requisitos e funcionalidades descrito na seção anterior, podemos delimitar que no banco devemos considerar os seguintes usuários: **Pylady**, **Representante de capítulo**, **Global Council**, **Participante** e **Visualizador externo**. Suas respectivas operações estão listadas abaixo:

- Visualizador externo

- Buscar eventos por região e data;
 - Dado um evento, consultar informações dele;
 - Realizar uma doação anônima;
 - Se cadastrar no sistema;
- Participante
 - Todas as operações de Visualizar externo;
 - Realizar doações com identificação;
 - Verificar doações passadas;
 - Se inscrever em eventos;
 - Ver histórico de eventos que participou;
 - Emitir certificados dos eventos que participou;
- Pylady
 - Ver e alterar os próprios dados;
 - Consultar informações do próprio e outros capítulos;
 - Ver histórico de atividades de qualquer capítulo;
 - Ver histórico de eventos de qualquer capítulo;
 - Ver histórico de colaborações de qualquer capítulo;
 - Buscar por outras Pyladies;
 - Ver informações do Global Council (gestão atual e passadas);
 - Ver histórico de eventos que apresentou;
 - Ver histórico de atividades que organizou;
 - Ver e alterar capítulo que participa;
- Representante de capítulo
 - Criar evento;
 - Criar atividade;
 - Excluir evento;
 - Excluir atividade;
 - Alterar dados dos eventos do próprio capítulo;
 - Alterar dados das atividades do próprio capítulo;
 - Designar Pyladies do próprio capítulo para atividades;

- Designar Pyladies do próprio capítulo para eventos;
 - Alterar dados de Pyladies do próprio capítulo;
 - Cadastrar colaborador;
 - Cadastrar colaboração;
 - Alterar informações do próprio capítulo;
- Global Council
 - Cadastrar projeto que participou;
 - Cadastrar presença em reunião;

2.3 Escolhas de implementação e possíveis problemas

Uma vez que o banco deve ser modelado maneira que esteja de acordo com as presentes práticas do PyLadies e atenda à suas atuais necessidades, algumas limitações devem ser consideradas e possíveis problemas devem ser tratados à nível de aplicação. Podemos citar:

- Possível dependência no ciclo **Pessoa - Pylady - Evento - Participante**:
Evento possui participação total de **Participantes** inscritos e **Pyladies** apresentadoras que geram, portanto, uma dependência, por isso é necessário tratar na implementação situações que inexista pyladies apresentadoras ou participantes inscritos, garantindo a integridade dos dados de registro de ambos.
- Possíveis ciclos de redundância nas interações entre **Pylady - Capítulo - Atividade - Participante - Evento** :
O sistema não garante que uma mesma Pylady se inscreva como participante em uma atividade (no caso de ser um **Evento**) que esta deva organizar, apresentar ou realizar (se esta estiver num **Capítulo**), afinal, subentende-se que ela já vai participar. Porém, esta redundância é necessária para possibilitar que pyladies possam participar de outros eventos sem que estejam necessariamente atribuídas a um capítulo, ou que não tiveram sua categorização como pylady validada para a execução das suas respectivas funções já que estes eventos são abertos.
- Possível redundância no ciclo **PyLady - Capítulo**: Não foi encontrada outra alternativa mais viável para atribuir uma representante de cada capítulo que não pelo **CR Representa**, dito isso, cogitou-se utilizar um atributo derivado, só que deixaria o sistema extremamente frágil e inseguro pois não garantiria a atribuição de representantes pyladies que não foram designadas às suas respectivas funções.

- Possível redundância no ciclo **Capítulos - Colaboração - Atividade**:

O sistema não garante que um mesmo colaborador faça um acordo ou um patrocínio que resulte numa atividade ao passo que também pode efetuar o mesmo processo com os **Capítulos** que também acaba resultando numa mesma atividade, portanto, tal redundância será tratada em implementação, pois esquematicamente é necessário assegurar tal semântica, pois colaborações não necessariamente precisam ser intermediadas por capítulos para efetuar atividades (sobretudo os **Projetos Colaborativos**).

- Possível dependência no ciclo **Pessoa - Doação - Capítulos**:

Para garantir a possibilidade de doações anônimas derivou-se a chave de **Capítulos** para a entidade fraca **Doação**, isso pode gerar uma dependência pelo fato de doações ainda assim poderem ser efetuadas pelas pessoas, mas que só poderiam ser destinadas a algo para serem registradas no banco a partir da dependência existencial de algum capítulo que deve ser atribuído a esta.

- Possível redundância no ciclo **Pylady - Mandato - PyladiesGlobalCouncil - Capítulos**

Pyladies podem ser integradas à um capítulo antes ou depois de assumirem um cargo no **PyladiesGlobalCouncil**, dito isso, mesmo que semanticamente seja redundante uma pylady participar ou deliberar sobre um capítulo tendo assumido este cargo ou não, o fato de poder ter múltiplos mandatos ou nenhum acaba fazendo com que tal redundância seja necessária, pois se fosse utilizado como um atributo, deixaria o sistema vulnerável à mudança de mandatos, renúncia, renovação e até num caso que uma pylady tenha que ser removida do sistema.

- Possível dependência no ciclo **Pyladies Colaboração - Colaboradores - Atividades - Projetos Colaborativos**

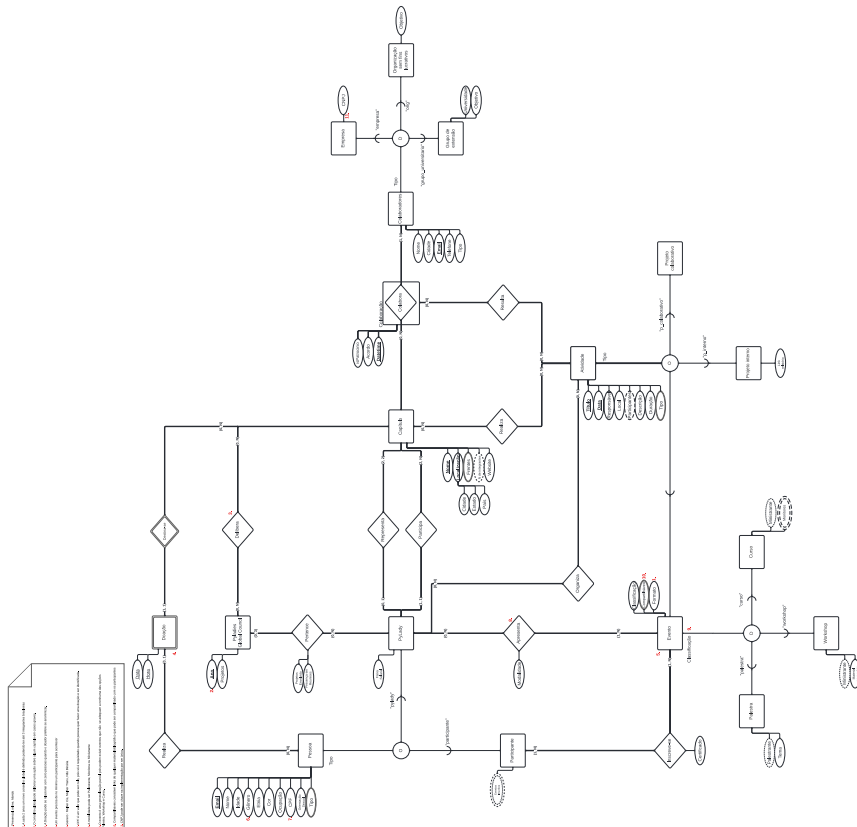
Semanticamente, não faz sentido um projeto designado como 'colaborativo' que não tivesse colaboradores, dito isso, **Colaboradores** possui relação parcial com **Atividades**, fazendo com que, de certa forma, por mais que a especialização permita, um projeto colaborativo sem colaboradores não seria possível, contudo, tal dependência se deve ao fato de que as próprias colaborações podem ser efetuadas de forma assíncrona, pois na organização de atividades o critério de início, fim e outros demais requisitos para a execução destas atividades ficam à rigor dos colaboradores a partir dos **Acordos**.

- Possível vulnerabilidade com múltiplos cadastros de uma mesma **Pessoa** caso ela tenha mais de um e-mail:

Uma vez que o PyLadies não coleta o número de documentos de identidade de seus membros (**Pylady**) nem de **Participante**, o uso de CPF ou similares como chave não é possível, abrindo a possibilidade de múltiplos cadastros. Pois o uso de e-mail

como identificador único de usuário é comum em outras aplicações, ele foi escolhido para ser usado como chave.

2.4 Diagrama do modelo entidade-relacionamento



3 Alterações realizadas na segunda parte do Projeto

Incorporaram-se explicações mais aprofundadas acerca dos ciclos e outras questões para serem documentadas no tópico 2.3 e incluiu-se as operações que serão feitas no banco no tópico 2.2, e, por fim, fez-se pequenos ajustes no MER para a maior corretude do esquemático e de conformidade com os padrões teóricos.



5 Escolhas de mapeamento

Por conta de particularidades do banco de dado e para garantir as relações como modeladas no MER, destacamos algumas escolhas feitas durante o mapeamento para o Modelo Relacional:

- **1. Relação N:M entre Participante e Evento**

Solução Adotada: No lugar de criar a tabela para Participante, para garantir a participação total de ambos os lados na relação Inscreve-se, foi feita apenas uma tabela ParticipaçãoEvento.

Vantagens: Garante a participação total de Participante no relacionamento Inscreve-se; facilita busca por Eventos Inscritos de dada Pessoa com o tipo Participante.

Desvantagens: Não temos uma tabela para obter diretamente os participantes, tendo que fazer uma consulta usando Pessoa e TipoPessoa; É necessário manter a consistência em aplicação para garantir que apenas Pessoas com o tipo Participantes se inscrevem em eventos.

Solução Alternativa: Mapear entidade Pessoa para sua própria tabela e usar de chave estrangeira para criar a tabela Inscreve-se.

- **2. Especialização de Atividade**

Solução Adotada: Para garantir a especialização total de Atividade, foi decidido fazer apenas as tabelas dos CEs Específicos, mesmo que isso tenha resultado na necessidade de criar diversas tabelas para atender aos relacionamentos Resulta, Realiza e Organiza.

Vantagens: Dados não são replicados e é garantida a especialização total de Atividade.

Desvantagens: É necessária a criação de uma tabela para cada um dos CEs específicos para atender às relações N:M Organiza, Realiza e Resulta. Também não há como usar chave estrangeira em TipoAtividade, sendo necessário garantir a consistência em aplicação.

Solução Alternativa: Criar uma tabela para Atividade e usar de chave estrangeira para os CEs específicos, enquanto trata especialização total em aplicação.

- **3. especialização de Evento**

Solução Adotada: No mapeamento de Evento, para que não haja problemas ao mapear os Monitores (atributo de Curso), foi decidido mapear cada um dos CEs em tabelas separadas.

Vantagens: É garantido a consistência de que todo Monitor está relacionado à um Evento da classificação Curso.

Desvantagens: Necessário criar uma tabela para cada classe de Evento, embora elas não participem de outros relacionamentos nem tenham muitos atributos próprios.

Solução Alternativa: Mapear tudo em uma tabela Evento e garantir em aplicação que só há monitores em Eventos da classe Curso.

- **4. Relação N:M entre PyLady e Evento**

Solução Adotada: Como os atributos Ministrante, Monitores e Palestrante, dos CEs específicos de Evento podem ser externos às PyLadies, estes não foram definidos como chave estrangeira. Entretanto, é necessário garantir em aplicação a coerência dos dados em relação ao atributo Modalidade do relacionamento Apresenta: todas PyLadies que apresentam um evento na modalidade Monitora, por exemplo, devem constar no atributo Monitores e assim por diante.

Vantagens: Simplifica o banco ao tratarmos palestrantes e ministrantes que não são PyLadies.

Desvantagens: Necessidade de tratar em aplicação a consistência com o relacionamento Apresenta.

Solução Alternativa: Separar os atributos Ministrante, Monitores e Palestrante em membro ou não do PyLadies, no qual nos membros são usadas chaves estrangeiras.

- **5. Relação N:M entre PyLady e Evento**

Solução Adotada: Na relação Apresenta, entre PyLady e Evento, não conseguimos garantir a participação total de Evento, portanto isso deve ser feito em aplicação.

Vantagens: Relacionamento N:M é garantido.

Desvantagens: A participação total deve ser garantida em aplicação.

Solução Alternativa: Nenhuma outra solução foi considerada.

- **6. Relação Representa 1:N entre PyLady e Capítulo**

Solução Adotada: Para garantir a participação total e a cardinalidade do relacionamento Representa entre PyLady e Capítulo, dois atributos foram adicionados em Capítulo, um para cada PyLady representante. Isso também garante a participação total no relacionamento Participa.

Vantagens: Todas as participações totais são garantidas.

Desvantagens: Não há desvantagens da solução nesse contexto.

Solução Alternativa: No mapeamento de PyLady, adicionar um atributo Representa. Deve ser garantido em aplicação que o capítulo tem duas representantes.

- **7. Especialização de Colaboradores**

Solução Adotada: Para garantir a exclusão mutua de Colaboradores, os CEs específicos Empresa, Organização sem fins lucrativos e Grupo de Extensão foram mapeados todos na mesma tabela. Isso foi feito como os CEs específicos não participam de relacionamentos, e terem poucos atributos próprios.

Vantagens: Não é necessário a criação de diversas tabelas para cada CE específico, ao mesmo tempo que garante a disjunção.

Desvantagens: Esperados diversos valores nulos em Colaboradores.

Solução Alternativa: Mapear cada CE específico em uma tabela, com a chave primária sendo uma chave estrangeira com Colaboradores.

- **8. Relação N:M entre Capítulo e Colaboradores**

Solução Adotada: Foi feita a tabela da agregação Colaboração, contendo o Capítulo e o Colaborador, bem como Datetime como chave. A participação total de Colaboradores no relacionamento deve ser garantida em aplicação.

Vantagens: Garante a cardinalidade do relacionamento e evita dados nulos e replicações.

Desvantagens: Não é possível garantir participação total de Colaboradores.

Solução Alternativa: Por conta da agregação bem como o relacionamento de Colaboração com Atividade, nenhuma outra solução foi considerada.

- **9. Atributos derivados**

- Palestrante - Mapeado no MR
- Ministrante - Mapeado no MR.
- Monitores - Mapeado no MR.
- Ministrante - Mapeado no MR.
- Eventos Inscritos - Atualizado na aplicação atualizado a partir da demanda.
- N de integrantes - Atualizado na aplicação atualizado a partir da demanda.
- Participantes - Atualizado na aplicação atualizado a partir da demanda.

6 Alterações feitas para a Parte 3

- **MER**

- **responsável** agora é um dado inserido e, portanto, um atributo simples monovalorado.

- Agregação **Mandato** removida pois não mais é necessária;

- **MR**

- Decidiu-se criar um ID sintético como chave primária em algumas entidades incremental para garantia de consistência e economia na alocação de muitas strings para serem replicadas ao longo do banco.
- Justificado os cálculos dos atributos derivados, que podem ser calculados sob demanda, e os outros que são mapeados normalmente e obtidos por junções.

7 Consultas

7.1 Consulta 1

Obter para cada pylady do capítulo de São Carlos seu nome, email e o número de eventos que ela foi responsável

```
1 SELECT P.nome, P.email, COUNT(*) FROM (Pylady A JOIN Pessoa B
   ON A.pessoa = B.id) P
2 JOIN Capitulo C ON C.id = P.capitulo
3 JOIN Evento E ON E.responsavel = P.id
4 WHERE C.cidade = 'São_Carlos'
5 GROUP BY P.nome,P.email;
6 -- Sa da esperada:
7 --
8 -- nome | email | count
9 -- -----+-----+-----
10 -- ana | ana@gmail.com | 2
11 -- clara | clara@gmail.com | 2
12 -- Para tal consulta de sele ão, utilizou-se operadores como
   GROUP BY para agrupar os usuários pelo nome uma vez que
   deseja-se exibir a quantidade de eventos de forma concisa.
```

7.2 Consulta 2

Obter nome do colaborador e as atividades resultantes de colaborações que não são com grupos universitários.

```
1 SELECT C.nome, F.titulo FROM
```



```

2      (SELECT A.nome, A.id as id_colaborador, B.id as
        colaboracao_id
3      FROM Colaborador A JOIN Colaboracao B ON B.colaborador
        = A.id
4      WHERE UPPER(A.tipo) <> 'GRUPO_UNIVERSITARIO') C
5  JOIN (SELECT colaboracao, titulo FROM ResultaEvento JOIN
        Evento ON ResultaEvento.evento = Evento.id
6  UNION ALL SELECT colaboracao, titulo FROM
        ResultaProjInterno JOIN ProjetoInterno ON
        ResultaProjInterno.PROJ_INTERNO = ProjetoInterno.id
7  UNION ALL SELECT colaboracao, titulo FROM
        ResultaProjColaborativo JOIN ProjetoColaborativo ON
        ResultaProjColaborativo.PROJ_COLABORATIVO =
        ProjetoColaborativo.id) F
8  ON C.colaboracao_id = F.colaboracao;
9  -- Sa da esperada:
10 --
11 --      nome      |      titulo
12 -----+-----
13 -- Colaborador 2 | curso aws
14 -- Colaborador 2 | curso azure

```

Para tal, uniu-se todos os conjuntos entidades para garantir que todas as tabelas envolvendo diferentes atividades possam ser atribuídas a um colaborador que se deseja exibir.

7.3 Consulta 3

Obter número total de participantes dos eventos organizados por cada pylady, seu nome e capítulo, ordenado pelo número de participantes

```

1  SELECT COUNT(*) AS participantes, P.nome AS pylady,
        Capitulo.nome AS capitulo
2  FROM (ParticipacaoEvento NATURAL JOIN OrganizaEvento) E
3  JOIN (Pylady JOIN Pessoa ON Pylady.pessoa = Pessoa.id) P ON
        E.pylady = P.id
4  JOIN Capitulo ON P.capitulo = Capitulo.id
5  GROUP BY P.id, P.nome, Capitulo.nome
6  ORDER BY COUNT(*) DESC;
7  -- Sa da esperada:
8  --
9  -- participantes | pylady | capitulo

```

```

10  -----+-----+-----
11  --          4 | ana      | São Carlos
12  --          1 | carolina | São Paulo

```

Usando group by para a exibição concisa dos dados para que se possa checar os respectivos eventos organizados por cada pylady possivelmente organizadora, e ordenando por quantidade de participantes para o tratamento de eventos em ordem de prioridade e impacto no contexto de grandiosidade.

7.4 Consulta 4

Obter média de faixa salarial das pyladies de cada capítulo, ordenado pela faixa salarial

```

1  SELECT Capitulo.nome, AVG(Pylady.faixa_salarial) AS
   Faixa_salarial_media FROM
2  Capitulo JOIN Pylady ON Pylady.capitulo = Capitulo.id
3  GROUP BY Capitulo.id, Capitulo.nome
4  ORDER BY AVG(Pylady.faixa_salarial) DESC;
5  -- Sa da esperada:
6  --
7  --     nome      | faixa_salarial_media
8  -----+-----
9  -- São Paulo    | 2200.0000000000000000
10 -- São Carlos   | 2000.0000000000000000

```

Utilizando agrupamento para uma exibição concisa e ordenando pela quantidade salarial para uma melhor visualização dos salário para efeitos comparativos de região para região (no contexto de precisar nivelar ou fazer reajustes de orçamentos).

7.5 Consulta 5

Obter total de atividades organizadas por cada capítulo e nome dos capítulos

```

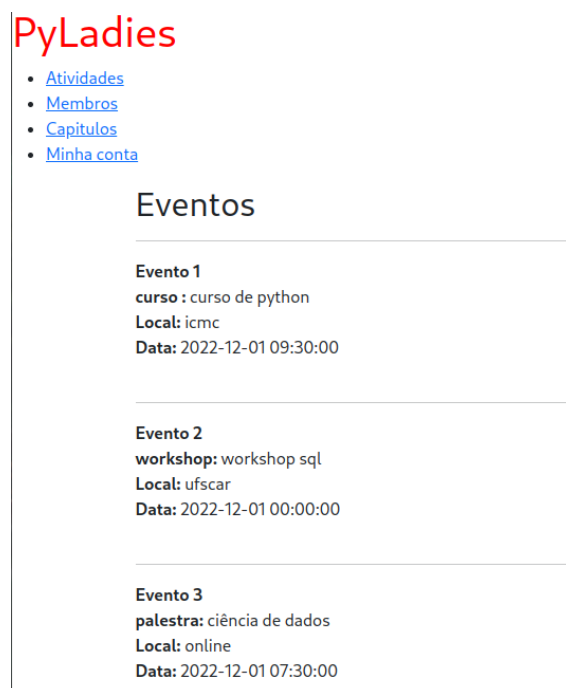
1  SELECT Capitulo.nome, COUNT(*) FROM
2  Capitulo JOIN (SELECT * FROM RealizaEvento
3  UNION ALL SELECT * FROM RealizaProjInterno
4  UNION ALL SELECT * FROM RealizaProjColaborativo) X ON
   X.capitulo = Capitulo.id
5  GROUP BY Capitulo.id, Capitulo.nome;

```

```
6 -- Sa da esperada:
7 --
8 --      nome      | count
9 -----+-----
10 -- ão Carlos'   |      7
11 -- São Paulo    |      3
```

Conseguir buscar por atividades e seus respectivos capítulos atribuídos, para tal foi crucial utilizar união das tabelas de atividades uma vez que estas possuem uma inconformidade de atributos, e para poder contabilizar e atribuir aos capítulos, foi necessário tais operações, agrupada para uma exibição mais concisa.

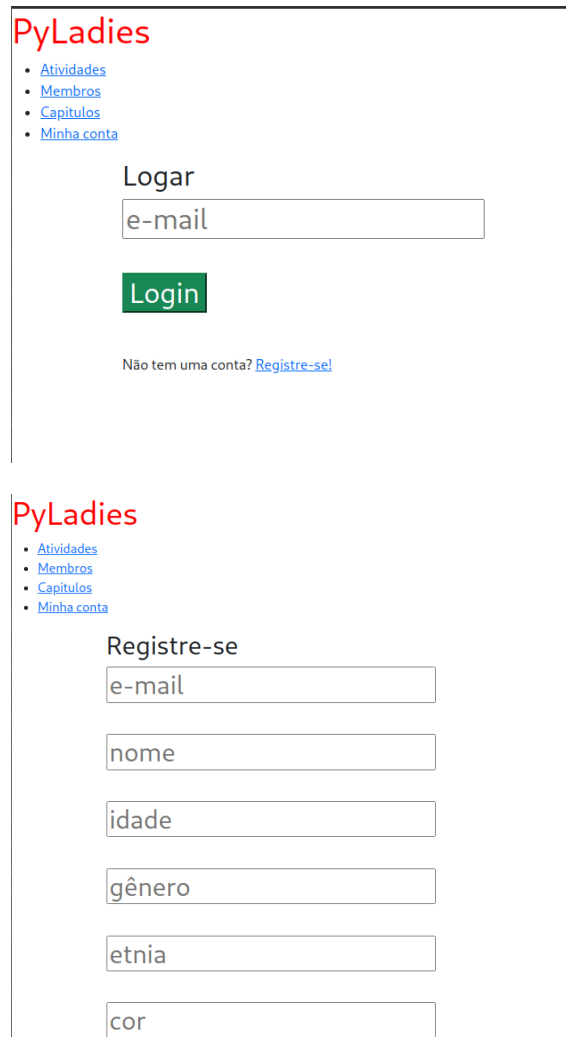
8 Descrição da Aplicação



A aplicação elaborada é um esboço de página web na qual uma pessoa externa ao PyLadies pode checar informações de eventos, capítulos e membros de grupo, bem como se cadastrar e ver suas próprias informações tais como certificados em atividades que participou.

O projeto foi elaborado usando a linguagem **Python** e o SGBD **PostgreSQL**, junto com as bibliotecas **Flask** para a elaboração do servidor e *psycopg2* para conexão e comunicação com o banco.

As buscas no banco são feitas para obter as atividades, pyladies, participantes e capítulos, enquanto a inserção é feita para cadastrar uma pessoa nova no banco.



The image displays two screenshots of the PyLadies website interface. The top screenshot shows the login page with the PyLadies logo, a list of links (Atividades, Membros, Capítulos, Minha conta), a 'Logar' label, an 'e-mail' input field, a green 'Login' button, and a link for 'Registre-se' for users without an account. The bottom screenshot shows the registration page with the same header and links, followed by a 'Registre-se' label and input fields for 'e-mail', 'nome', 'idade', 'gênero', 'etnia', and 'cor'.

9 Conclusão

O tema que começou baseado num modelo com uma lógica de negócio real e praticável. A tradução da ideia inicial do projeto para o Modelo de Entidades e Relacionamentos (MER) foi um grande susto devido às inúmeras soluções propostas. Contudo, na solução final, a partir da aplicação real do modelo de negócio do grupo PyLadies, fez com que o MER se tornasse um dos maiores desafios que tivemos durante todo o trabalho porque foi o primeiro instante que tivemos que pensar na aplicação abstrata e mais na prática, a partir do uso concreto com as limitações semânticas das modelagens. Em seguida, o Modelo Relacional se mostrou inicialmente fácil pois o MER por si só já simplificou a aplicação abstrata. No entanto, houveram algumas discussões sobre como poderíamos organizar o diagrama de forma que fosse mais eficiente tanto em memória, como aplicação. Durante a construção do banco de dados, com a criação de esquema e inserção de dados, não houve tanto trabalho. Nesse momento o principal problema foi a necessidade de gerar o contingente de dados para tornar as consultas eficientes. Logo no processo de criação do script de consultas acabou sendo bem proveitoso. Foi necessário

9. CONCLUSÃO

pensar em consultas que fariam sentido no contexto da aplicação mas que também fosse algo desafiador de otimizar. Para nós a maior dificuldade na sanitização dos dados mais sensíveis como no caso de email e cnpj, embora a utilização de uma interface gráfica ao invés de usar o próprio terminal agregou mais desafios. Por fim, a possibilidade de utilizar os componentes da forma correta, é necessário pensar de modo que a modelagem faça sentido, e isso se mostrou muito interessante para o grupo.