**Regras de mapeamento**

**Modelagem GitHub**

Disciplina: Fundamentos de Banco de Dados

Grupo: Nicolle Favero e Matheus Azambuja

Professora: Karin Becker

**Mapeamento de Entidades**

Com base na modelagem conceitual e nas especificações no dicionário de dados decidimos optar pelas seguintes regras.

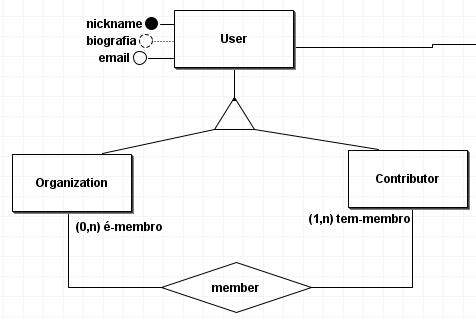
Para melhor apresentação das regras seguiremos a seguinte sequência de transformação do modelo ER em um modelo relacional:

* Entidades e respectivos atributos;
* Generalizações/especializações;
* Relacionamentos e respectivos atributos;

Notações:

* **Chave primário em negrito;**
* *Chave estrangeira em itálico;*
* Atributo opcional destacado com \*;
* Atributo único destacado com sublinhado;

**Entidade User: especialização**



Considerando a especialização que a entidade User possui, as distintas relações que Organization e Contributor participam e pelas operações que serão realizadas nessas tabelas decidimos transformar a especialização nas seguintes tabelas:

(para melhorar as relações das tabelas, criamos um atributo ID. Esse atributo será usado no lugar do nickname, que se transformará em uma chave alternativa).

Users (**id**, name\*, nickname, biografia\*, email);

Contributors (**id**, *idUser*)

*references Users (idUsers);*

Organizations (**id**, *idUser*)

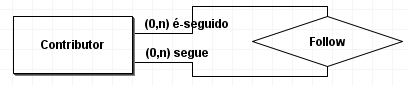
*references Users (idUsers);*

Members (***idContrib***, ***idOrg***)

*references Contributor(idContrib),*

*references Organization(idOrg);*

**Auto Relacionamento Follow:**



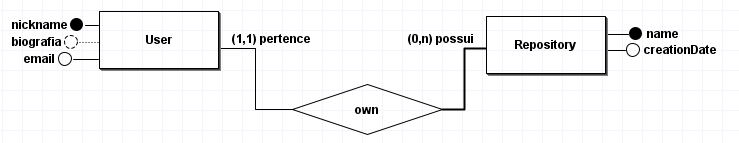
Criado uma tabela para o relacionamento por se tratar de um Relacionamento N-M:

Follows (***idFollower***, ***idFollowed***, date)

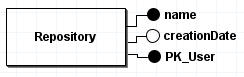
references Contributors(IdFollower),

references Contributors(idFollowed);

**Entidade Repository**



Mapeado para:



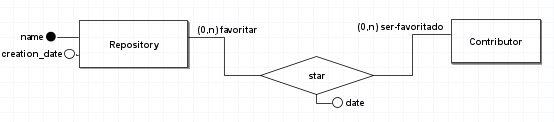
Esse mapeamento foi feito considerando a resolução do Relacionamento 1-N e a Entidade Fraca. Dessa forma, cada repositório será único dentre os repositórios do usuário.

Repositories (**idRepo**, ***idUser***, name, creationDate)

*references Users (idUser);*

Escolhemos essa resolução para tabela Repository pois ela participará de muitos relacionamentos posteriormente (Dar exemplos).

**Relação Start**



Mapeado para:

Stars (***idContrib***, ***idRepository***, date)

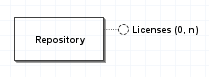
*references Contributor (idContrib),*

*references Repository (idRepository);*

Relacionamento N-M, mapeamento feito seguindo a padronização feita segundo aula, pois há somente uma alternativa para tal. Por queremos representar uma timeline e esse relacionamento participa dela, essa é a justificativa para modelarmos dessa forma.

**Entidade License e relação com Repository (ATRIBUTO MULT-VALOR?)**

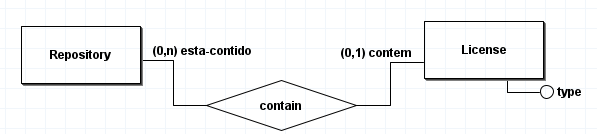
**(N-M talvez??)**

****

Mapeado para:

Licenses (***idRepository***, **cod,** type)

*references Repository (idRepository);*



Mapeado para:

License (**id**, type)

Repositories (**idRepository**, *idUser*, name, creationDate, *idLicense*)

*references Users (idUser),*

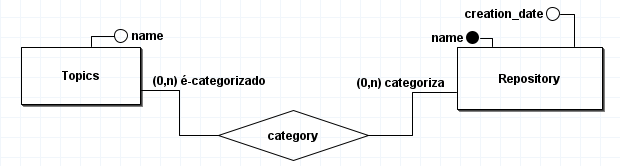
*references Licence (idLicense);*

Alterações feitas:

* Repository recebeu novo atributo: IdLicense

Decidimos por esse mapeamento pois o Relacionamento é 1-N, portanto, representando assim, o desejado.

**Entidade Topics e relação com Repository**



Criado uma tabela para o relacionamento por se tratar de um Relacionamento N-M:

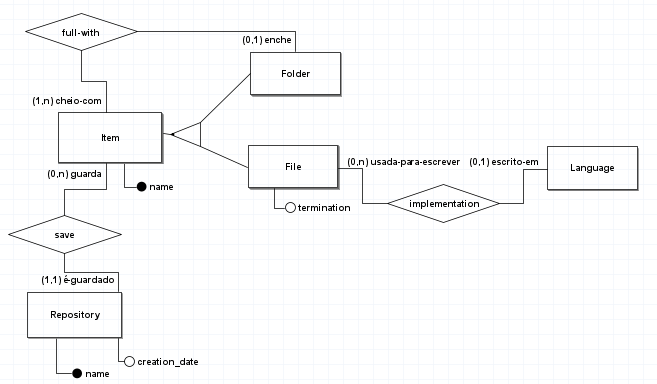
Topics (**cod**, name)

Category (***idRepository***, ***codTopic***)

*references Repository (idRepository),*

*references Topics (codTopic);*

**Estruturação do repositório**



**Entidade Item: especialização e relação com Repository**

Mapeado para:

Items (**id**, name, *idRepository*)

*references Repository (idRepository);*

Folders (**idFolder**, *idItem*)

*references Items (idItem);*

Files (**id**, *idItem*, termination)

*references Item (idItem);*

Por se tratar de um Relacionamento 1-N (Obrigatório do “lado 1” / Opcional do “lado N”), mapeamos dessa maneira. Optamos por essa alternativa considerando o relacionamento de Item com Repository. Além disso, consultar quantos arquivos um repositório tem ou quantas pastas ele possui fica trivial.

FullWith (***idItem***, *idFolder*)

*references Items (idItem),*

*references Folders (idFolders);*

O Relacionamento 1-N (Opcional do “lado 1” / Opcional do “lado N”) FullWith foi modelado dessa forma para facilitar a consulta posteriormente para obter quantos arquivos uma pasta contém. Assim sendo, fica fácil realizar tal consulta.

**Entidade Language:**

Mapeado para:

Languages (**cod**, name)

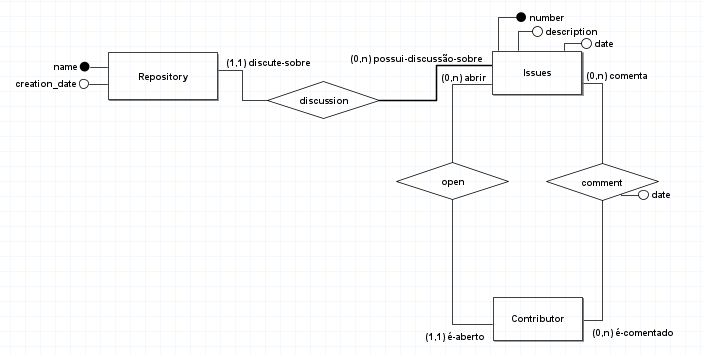
Implementations (***idFile,*** *codLanguage*)

*references Files (idFile),*

*references Languages (codLanguage);*

Utilizamos esse mapeamento pensando no futuro: posteriormente será implementado uma consulta importante com o objetivo de obter o percentual das linguagens utilizadas em um repositório. A relação entre Files e Languages modelado dessa forma, visa aumentar o desempenho dessa consulta em específico.

**Entidade Issue e relações**



**Relação Discussion**

Mapeado para:

Issues (**number**, description, date, ***idRepository***)

*references Repositories (idRepository);*

Esse mapeamento foi feito considerando a resolução do Relacionamento 1-N e a Entidade Fraca. Dessa forma, cada Issue será única dentre as Issues do repositório.

**Relação Open**

Mapeado para:

Issues (**number**, description, date, *idRepository, idContrib*)

*references Repositories (idRepository),*

*references Contributor (idContributor);*

Adicionado o atributo (idContrib) que representará o contribuidor que abriu o Issue, representando assim o relacionamento.

**Relação Comment**

Mapeado para:

Comment (***idContrib***, ***numIssue***, **date**, description)

*references Contributors (idContrib),*

*references Issues (numIssue);*

Relacionamento N-M, mapeamento feito seguindo a padronização feita segundo aula, pois há somente uma alternativa para tal. Por queremos representar uma timeline e esse relacionamento participa dela, essa pode ser uma justificativa para modelarmos dessa forma.