Universidade Estadual de Campinas - Instituto de Computação

MC536 - Bancos de Dados - Profª. Claudia Bauzer Medeiros

**Trabalho 2**  
*agência matrimonial GetLove*

Bruno Sepulveda (165207) **·** Giovanni Bertão (173325) **·** Guilherme Gama (173608)  
Guilherme Perrotta (173846) **·** Gustavo Salibi (174135) **·** João Pedro Martins (176117)

**ESQUEMA**

**ENTIDADES DO MODELO ER:**

**Pessoa**(id\_pessoa, cpf, email, nome, data\_nascimento, sexo, cidade, uf, numero\_filhos, tabaco, alcool, outras\_drogas, religiao, tipo\_de\_personalidade, latitude, longitude)

Dependências funcionais:

id\_pessoa → cpf, email, nome, data\_nascimento, sexo, cidade, uf, numero\_filhos, tabaco, alcool, outras\_drogas, religiao, tipo\_de\_personalidade, latitude, longitude)

Dependências de inclusão:

id\_pessoa deve ser não-nulo e único na coluna.

cpf, email, nome, data\_nascimento, cidade, uf, sexo devem ser não-nulos.

**Moderador**(id\_moderador, nome)

Dependência funcional:

id\_moderador → nome

Dependência de inclusão:

id\_moderador deve ser não-nulo e único na coluna.

nome deve ser não-nulo

**Ocorrencia**(id\_ocorrencia, tipo, id\_implicado, ref\_arquivo\_mensagem, data\_ocorrencia)

Dependências funcionais:

id\_ocorrencia → tipo, id\_implicado, ref\_arquivo\_mensagem, data\_ocorrencia

Dependências de inclusão:

id\_ocorrencia deve ser não-nulo e único na coluna.

id\_implicado, ref\_arquivo\_mensagem, data\_ocorrencia devem ser não-nulos

**Empresa**(id\_empresa, cnpj, nome, email, telefone, cidade, uf)

Dependências funcionais:

id\_empresa → cnpj, nome, email, telefone, cidade, uf

Dependências de inclusão:

id\_empresa deve ser não-nulo e único na coluna.

**Empresa\_de\_dados**(id\_empresa)

Dependências funcionais:

não há

Dependências de inclusão:

id\_empresa deve ser chave de alguma tupla em **Empresa**.

**Empresa\_de\_servicos**(id\_empresa)

Dependências funcionais:

não há

Dependências de inclusão:

id\_empresa deve ser chave de alguma tupla em **Empresa**.

**RELACIONAMENTOS NO MODELO ER:**

**Combina\_com**(id\_pessoa1, id\_pessoa2, ref\_arquivo\_conversa, data\_combinacao)

Dependências funcionais:

id\_pessoa1, id\_pessoa2 → ref\_arquivo\_conversa, data\_combinacao

Dependências de inclusão:

A chave composta id\_pessoa1, id\_pessoa2 deve ser não-nula e única na coluna.

ref\_arquivo\_conversa e data\_combinacao devem ser não-nulos

**Aprova**(id\_pessoa, id\_aprovado, data\_aprovacao)

Dependências funcionais:

id\_pessoa1, id\_pessoa2 → data\_aprovacao

Dependências de inclusão:

A chave composta id\_pessoa1, id\_pessoa2 deve ser não-nula e única na coluna.

data\_aprovacao deve ser não-nula

**Tem\_dados\_vendidos\_a**(id\_pessoa, id\_empresa)

Dependências de inclusão:

A chave composta id\_pessoa1, id\_empresa deve ser não-nula e única na coluna.

**Abre**(id\_ocorrencia, id\_pessoa)

Dependências funcionais:

id\_ocorrencia → id\_pessoa

Dependência de inclusão:

id\_ocorrencia deve ser não-nulo e único na coluna

**Avalia**(id\_ocorrencia, id\_moderador)

Dependências funcionais:

id\_ocorrencia → id\_moderador

Dependência de inclusão:

id\_ocorrencia deve ser não-nulo e único na coluna

**ENTIDADES FRACAS NO MODELO ER:**

**Plano\_premium**(id\_pessoa, data\_expiracao)

Dependências funcionais:

id\_pessoa → data\_expiracao

Dependências de inclusão:

id\_pessoa deve ser chave de tupla na tabela Pessoa

**Servico**(id\_empresa, numero\_servico, preco, categoria, ref\_arquivo\_descricao, latitude, longitude)

Dependências funcionais:

id\_empresa, numero\_servico → preco, categoria, ref\_arquivo\_descricao, latitude, longitude

Dependências de inclusão:

id\_empresa deve ser chave de tupla na tabela Empresa\_de\_Servicos

a chave composta (id\_empresa, numero\_servico) deve ser não-nula e única na coluna

**Foto**(id\_pessoa, numero\_foto, ref\_arquivo\_imagem)

Dependências funcionais:

id\_pessoa, numero\_foto → ref\_arquivo\_imagem

Dependências de inclusão:

id\_pessoa deve ser chave de tupla na tabela Pessoa

a chave composta (id\_pessoa, numero\_servico) deve ser não-nula e única na coluna

ref\_arquivo\_imagem deve ser única na coluna e não-nula

**Imagem\_servico**(id\_empresa, numero\_servico, numero\_imagem, ref\_arquivo\_imagem)

Dependências funcionais:

id\_empresa, numero\_servico, numero\_imagem → ref\_arquivo\_imagem

Dependências de inclusão:

a chave parcial (id\_empresa, numero\_servico) deve ser chave de tupla na tabela Servico

a chave composta (id\_empresa, numero\_servico, numero\_imagem) deve ser não-nula e única na coluna

**RELACIONAMENTOS FRACOS NO MODELO ER:**

***implementadas em SQL por chave estrangeira para o dono (após minimização).***

É assinado(id\_pessoa)

É imagem de(id\_produto, numero\_imagem)

Tem\_foto(id\_pessoa, numero\_foto)

Oferece(id\_empresa, numero\_servico)

**Lista de operações**

**(especificação completa das operações foi refeita no T1 corrigido.)**

**Consultas:**

Encontrar pessoas

Gerar encontro

Listar combinações

Listar quem me aprovou

Recuperar log de conversa

Listar ocorrências abertas por cliente x:

Listar ocorrências sobre cliente x:

Listar ocorrências sobre a empresa x:

Listar ocorrências sobre a getLove:

Listar ocorrências abertas:

**Atualizações:**

Remover combinação

Alterar perfil

Refazer teste de personalidade

Recalcular posição geográfica

Alterar fotos

Alterar plano

Alterar lista de empresas parceiras

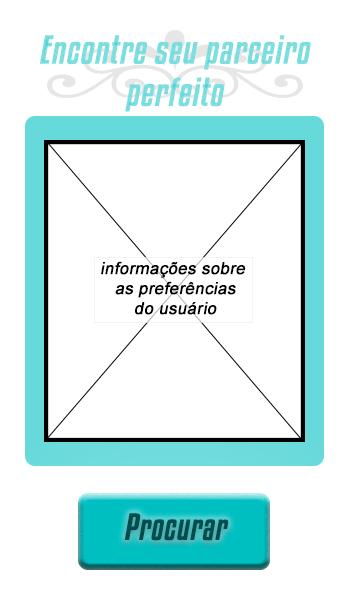
Alterar lista de produtos

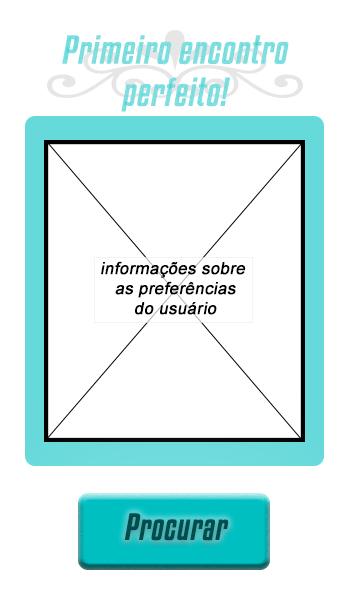
Alterar imagem de produto

**Questões de implementação**

**1**. Optamos por demonstrar o uso do banco de dados utilizando comandos SQL.

A demonstração se dará através de uma plataforma fictícia, onde o sistema executaria funcionalidades específicas além dos comandos SQL, tais como: fazer combinação entre pessoas, exibir ao usuário quem demonstrou interesse por ele, selecionar melhor opção de encontro a ser recomendado pela plataforma, funcionalidade de escrever e enviar mensagens, uma tela para denunciar outros usuários, além de possibilidade de atualizar perfil alterando informações. Seguem abaixo algumas imagens que ilustram como seria tal plataforma:

**1.  2.  3.**

**4.5.6.**

**2**. O banco de dados será criado utilizando os seguintes comandos do SGBD MySQL:

|  |
| --- |
| **create** **database** GetLove; **use** GetLove;  **create** **table** Pessoa (  id\_pessoa int primary **key**,  cpf char(11) **not** null **unique**,  email varchar(30) **not** null **unique**,  nome varchar(100) **not** null,  data\_nascimento date **not** null,  sexo char **not** null,  cidade varchar(30) **not** null,  uf char(2) **not** null,  numero\_filhos int,  tabaco int,  alcool int,  outras\_drogas int,  religiao int,  tipo\_de\_personalidade int,  latitude float(10,6),  longitude float(10,6) );  **create** **table** Moderador (  id\_moderador int primary **key**,  nome varchar(100) **not** null );  **create** **table** Ocorrencia (  id\_ocorrencia int primary **key**,  tipo char(7) **not** null,  id\_implicado int,  ref\_arquivo\_mensagem varchar(50) **not** null,  data\_ocorrencia date **not** null,  **check** (tipo **like** "getlove" **or** tipo **like** "usuario" **or** tipo **like** "empresa") );  **create** **table** Empresa (  id\_empresa int primary **key**,  cnpj char(14) **not** null **unique**,  nome varchar(100) **not** null,  email varchar(30) **not** null **unique**,  telefone varchar(11) **not** null **unique**,  cidade varchar(30) **not** null,  uf char(2) **not** null );  **create** **table** Empresa\_de\_dados (  id\_empresa int **not** null,  foreign **key**(id\_empresa) **references** Empresa(id\_empresa) );  **create** **table** Empresa\_de\_servicos (  id\_empresa int **not** null,  foreign **key**(id\_empresa) **references** Empresa(id\_empresa) );  **create** **table** Combina\_com (  id\_pessoa1 int,  id\_pessoa2 int,  ref\_arquivo\_conversa varchar(50) **not** null,  data\_combinacao date **not** null,  primary **key**(id\_pessoa1, id\_pessoa2) );  **create** **table** Aprova (  id\_pessoa int,  id\_aprovado int,  data\_aprovacao date **not** null,  primary **key**(id\_pessoa, id\_aprovado) );  **create** **table** Tem\_dados\_vendidos\_a (  id\_pessoa int,  id\_empresa int,  primary **key**(id\_pessoa, id\_empresa) );  **create** **table** Abre (  id\_ocorrencia int,  id\_pessoa int,  primary **key**(id\_ocorrencia) );  **create** **table** Avalia (  id\_ocorrencia int,  id\_moderador int,  primary **key**(id\_ocorrencia) );  **create** **table** Plano\_premium (  id\_pessoa int **not** null,  data\_expiracao date **not** null,  foreign **key**(id\_pessoa) **references** Pessoa(id\_pessoa) );  **create** **table** Servico (  id\_empresa int **not** null,  numero\_servico int **not** null,  preco float(10,2),  categoria varchar(5),  ref\_arquivo\_descricao varchar(50),  latitude float(10,6),  longitude float(10,6),  foreign **key**(id\_empresa) **references** Empresa\_de\_servicos(id\_empresa),  primary **key**(id\_empresa, numero\_servico) );  **create** **table** Foto (  id\_pessoa int **not** null,  numero\_foto int **not** null,  ref\_arquivo\_imagem varchar(50) **not** null **unique**,  foreign **key**(id\_pessoa) **references** Pessoa(id\_pessoa),  primary **key**(id\_pessoa, numero\_foto) );  **create** **table** Imagem\_servico (  id\_empresa int **not** null,  numero\_servico int **not** null,  numero\_imagem int **not** null,  ref\_arquivo\_imagem varchar(50) **not** null,  foreign **key**(id\_empresa, numero\_servico) **references** Servico(id\_empresa, numero\_servico),  primary **key**(id\_empresa, numero\_servico, numero\_imagem) ); |

**3.**

**a)**O banco de dados será alimentado pela importação de arquivos csv. O conteúdo desses arquivos virá de python scripts criados por nós. Esses scripts devem gerar grande quantidade de entidades aleatórias com atributos advindos de outros bancos de dados (como o atributo nome, de Pessoa) e alguns gerados aleatoriamente (como os atributos numéricos) para criar os arquivos .csv.

**b)** A tabela 1 lista o número de tuplas esperadas para cada relação .

**c)** Dificuldades imaginadas na implementação:

* Calcular distância entre clientes ou entre clientes e produtos, no intuito de alocar ou sugerir serviços que estejam próximos aos usuários envolvidos.
* Encontrar bancos de dados grandes para download que forneçam os dados necessários para geração das entidades que serão utilizadas para alimentar o banco de dados da aplicação, mencionadas em (a).
* Criar os python scripts para gerar os arquivos .csv mencionados em (a), com várias tuplas de todas as entidades, que possuem muitos atributos cada e com bastante variação de tipos.

|  |  |
| --- | --- |
| **Tabela 1 - Listagem da quantidade prevista de tuplas para cada relação criada** | |
| **Relação** | **Quantidade prevista (lim. superior)** |
| Pessoa | 500 |
| Ocorrencia | 50 |
| Moderador | 10 |
| Empresa | 50 |
| Plano\_premium | 200 |
| Servico | 100 |
| Foto | 1000 |
| Imagem\_servico | 200 |
| Combina\_com | 1250 |
| Aprova | 2500 |
| Tem\_dados\_vendidos\_a | 5000\* |
| Eh\_assinado | 200 |
| Eh\_imagem\_de | 200 |
| Tem\_foto | 1000 |
| Oferece | 100 |
| Abre | 50 |
| Avalia | 25 |
| \*: Diversos fatores podem influenciar esse valor, uma vez que qualquer tipo de dado pode ser vendido a qualquer empresa interessada. Contudo, pela natureza do programa consideramos os seguintes dados: Produtos recém adquiridos e produtos visualizados. Tais dados, se baseiam em uma aproximação inferior diferente das outras linhas que estão relacionadas ao limite superior. | |

**Mudanças de rumo**

Serviço Casamento dos Sonhos foi removido para focarmos nos serviços essenciais, então as queries “adicionar noivado” e “gerar cerimônia de casamento” também foram removidas.

Primeiro Encontro Perfeito e o serviço de encontrar parceiros em potencial agora levam a distância em consideração, necessitando que a aplicação grave dados de coordenada geográfica no banco de dados.