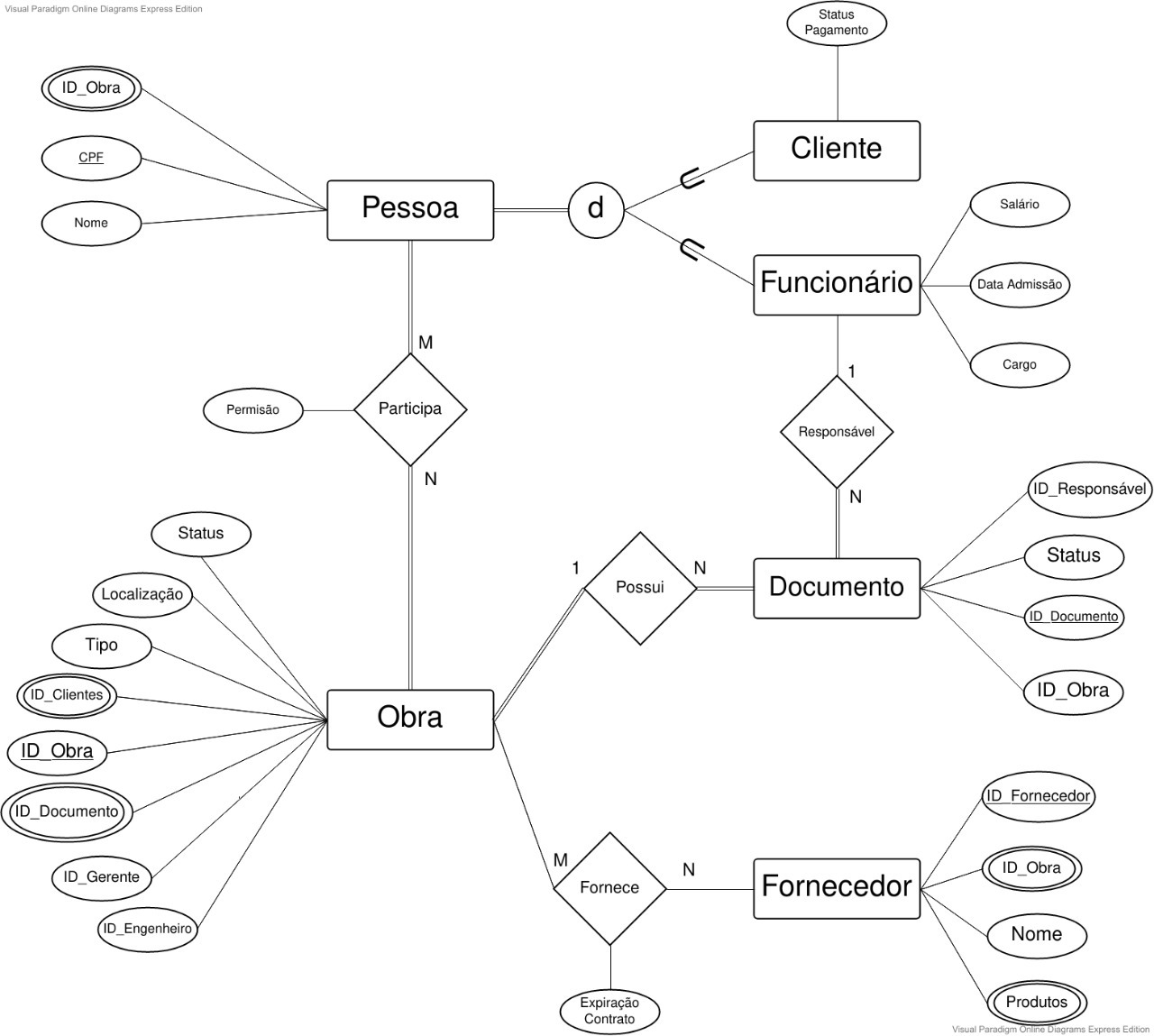
UNIVERSIDADE FEDERAL DE MINAS GERAIS

DISCIPLINA: INTRODUÇÃO A BANCO DE DADOS - DCC011

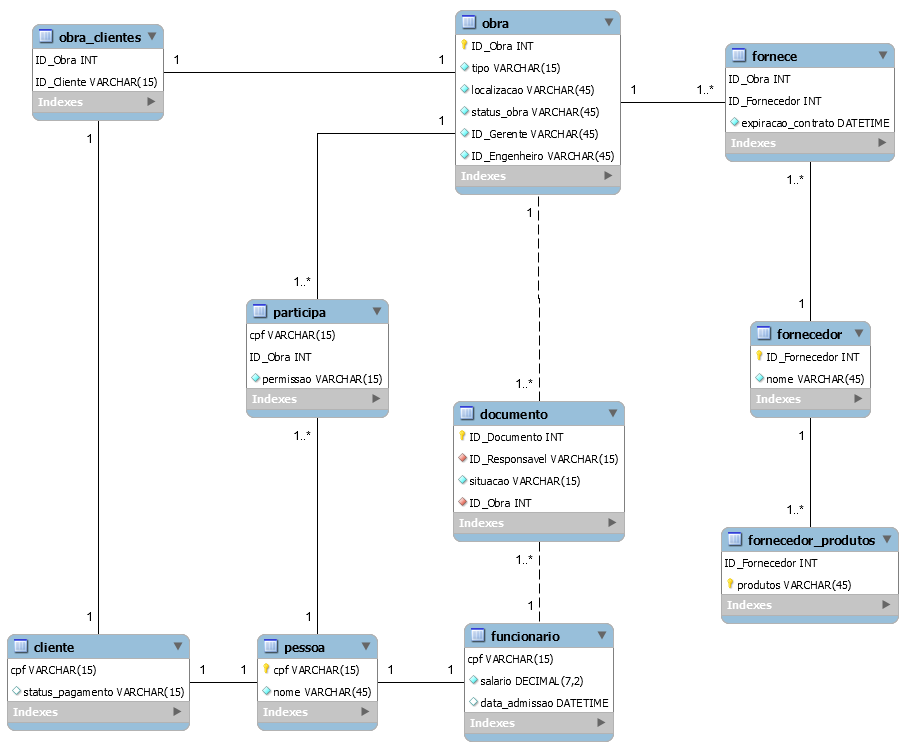
TRABALHO PRÁTICO 2

Integrantes:

Leandro Luiz Duarte Teixeira



Projeto Lógico



Este é o projeto lógico que fizemos a partir do Diagrama Entidade-Relacionamento apresentado acima. A modelagem do fornecedor foi a mais objetiva; ele possui um ID e um nome na própria tabela. Para implementar os produtos, foi usada uma tabela fornecedor-produtos. Foi cogitado também criar uma tabela só para os produtos e fazer disso uma relação M x N. Essa ideia, porém, foi descartada; fazer isso exigiria uma padronização da forma de apresentação dos produtos.

Da forma como implementamos, um fornecedor está livre para fornecer “cimento” enquanto um outro pode optar por fornecer apenas “cimento da marca Mauá”, sem exigir que o primeiro modifique os dados para apresentar todas as marcas e variedades que ele oferta ou que o segundo generalize o produto mesmo quando ele só vende uma marca específica que pode não ser a desejada para aquela obra específica.

A variável multivalorada “Id\_Obra” presente em Fornecedor pode ser obtida implicitamente através da tabela Fornece, que representa a relação entre o fornecedor e a obra. Ao inserir as restrições de modificação na tabela Fornece, foi decidido inserir bloqueio a mudança e cascata a remoção em ambas as chaves. A lógica é, uma vez firmado um contrato entre as duas partes e inserida a data de expiração, nenhuma outra relação pode vir a herdar essa data de expiração sem o firmamento de um novo contrato. Por fins de segurança, se for necessário trocar o ID de um fornecedor ou de uma obra, os contratos previamente definidos deverão ser cancelados e refirmados. Por outro lado, se um fornecedor ou obra deixa de existir, todos os contratos a eles relacionados podem ser removidos.

Ao modelar a obra, foi levado muito em consideração as potenciais exigências do mundo real também. Além dos dados univalorados que são mais objetivos, toda obra deve ter ao menos um documento. Porém, inserir uma restrição de integridade forçaria os usuários do banco de dados a só inserirem uma obra no sistema uma vez que seu documento tenha sido registrado. Nós achamos melhor permitir que a obra apareça no sistema no momento em que ela tem um engenheiro e um gerente já registrados. Por outro lado, os documentos só podem ser inseridos no sistema se possuírem uma obra correspondida. E, seguindo a mesma linha do fornecedor-obra, não é possível modificar a chave de um responsável ou obra enquanto estes tiverem documentos atribuídos a eles. É possível remover uma obra com documentos, isso gerará uma reação em cascata, porém não é possível remover um funcionário responsável por documento enquanto ele ainda possuir documentos atribuídos

Situação parecida acontece com a relação obra-cliente. Uma forma alternativa de implementar seria inserir o Id da obra como chave estrangeira na tabela cliente, visto que todo cliente deve possuir uma obra relacionada (caso contrário, não deve ser considerado cliente ainda). Todavia, optamos por criar uma tabela obra-clientes justamente porque ela permite aos usuários do sistema inserir um cliente que ainda está escolhendo a obra que deseja comprar. Outra coisa importante para se notar sobre a relação obra-cliente é que há um bloqueio a remoção inserido apenas no ID\_Obra. Um cliente pode decidir desistir de uma obra a qualquer momento sem dar uma satisfação direta às respectivas obras. Porém, antes de deletar uma obra, é importante garantir que todos os seus clientes tenham sido removidos ou atribuídos a outras obras.

Consultas

Na hora de formular as consultas, buscamos pensar em um conjunto que conseguisse cobrir da melhor forma possível as relações, tentando capturar as informações singulares que podem ser obtidas através de diversas tabelas.

De um ponto de vista do SQL, também tentamos selecionar ao menos uma query para testar cada uma das seguintes hipóteses:

* Se é mais barato (computacionalmente falando) fazer um join de forma explícita entre duas tabelas ou pesquisar, através da chave estrangeira, qual elemento da segunda tabela atende aos requisitos e através disso encontrar o dado correspondente na primeira tabela
* Se, para eliminar dados repetidos, é mais eficiente usar group by ou distinct.
* Se, para descobrir o maior valor em uma tabela, é melhor usar max ou colocar os dados em ordem decrescente e limitar a pesquisa para exibir apenas um dado.
* Se, para descobrir dados que estão em uma tabela e não estão na outra, é melhor usar not in ou fazer um left join na primeira e selecionar apenas os dados em que a chave estrangeira tem valor nulo.
* Os impactos potenciais de colocar uma subquery após a palavra chave from.
* Se é mais eficiente procurar um dado através da seleção de dados que atendem à condição desejada ou através da procura dos dados que não estão entre os que não atendem a essa condição.

Levando esses pontos em consideração, as queries escolhidas foram, agrupadas pelo principal critério que satisfazem, as seguintes:

Apenas seleção e projeção (2)

1 – Selecionar os CPFs dos funcionários que ganham entre 2000 e 3000 reais.

*SELECT CPF from Funcionario WHERE salario > 2000 AND salario < 3000*

*SELECT CPF from Funcionario WHERE CPF in (SELECT CPF from Funcionario WHERE salario > 2000) AND CPF in (SELECT CPF from Funcionario WHERE salario < 3000)*

2 – Selecione os status dos sobrados que serão construídos em Belo Horizonte.

*SELECT status\_obra FROM Obra WHERE tipo = 'Sobrado' and localizacao = 'Belo Horizonte'*

*SELECT status\_obra FROM Obra WHERE ID\_Obra IN (SELECT ID\_Obra FROM Obra WHERE tipo = 'Sobrado') AND ID\_Obra IN (SELECT ID\_Obra from Obra WHERE localizacao = 'Belo Horizonte')*

Junção de duas relações (3)

3 – Selecionar o nome de todas as empresas que fornecem telhas.

*SELECT nome FROM Fornecedor a JOIN Fornecedor\_Produtos b ON a.ID\_Fornecedor = b.ID\_Fornecedor WHERE produtos = 'Telha'*

*SELECT nome FROM Fornecedor WHERE ID\_Fornecedor IN (SELECT ID\_Fornecedor FROM Fornecedor\_Produtos WHERE produtos = 'Telha')*

4 – Selecionar os nomes dos funcionários estão há ao menos três anos na empresa.

*SELECT nome FROM Funcionario a JOIN Pessoa b ON a.CPF = b.CPF WHERE CURRENT\_TIMESTAMP - data\_admissao >= 3*

*SELECT nome FROM Pessoa a JOIN Funcionario b ON a.cpf = b.cpf WHERE a.cpf NOT IN (SELECT cpf FROM Funcionario WHERE CURRENT\_TIMESTAMP - data\_admissao < 3)*

5 – Selecionar o nome e os produtos do fornecedor que fornece a maior quantidade de produtos.

*SELECT nome, produtos FROM Fornecedor a JOIN Fornecedor\_Produtos b on a.ID\_Fornecedor = b.ID\_Fornecedor WHERE a.ID\_Fornecedor IN (SELECT ID\_Fornecedor FROM (SELECT a.ID\_Fornecedor, COUNT(produtos) as qtd FROM Fornecedor\_Produtos a JOIN Fornecedor b ON a.ID\_Fornecedor = b.ID\_Fornecedor GROUP BY a.ID\_Fornecedor) as q1 WHERE qtd in (SELECT max(qtd) FROM (SELECT a.ID\_Fornecedor, COUNT(produtos) as qtd FROM Fornecedor\_Produtos a JOIN Fornecedor b ON a.ID\_Fornecedor = b.ID\_Fornecedor GROUP BY a.ID\_Fornecedor)as q1))*

*SELECT nome, produtos FROM Fornecedor a JOIN Fornecedor\_Produtos b on a.ID\_Fornecedor = b.ID\_Fornecedor WHERE a.ID\_Fornecedor IN (SELECT ID\_Fornecedor FROM (SELECT a.ID\_Fornecedor, COUNT(produtos) as qtd FROM Fornecedor\_Produtos a JOIN Fornecedor b ON a.ID\_Fornecedor = b.ID\_Fornecedor GROUP BY a.ID\_Fornecedor ORDER BY qtd DESC LIMIT 1) as q1)*

Junções de três ou mais relações (3)

6 – Selecionar os status de pagamento de todas as obras localizadas em Belo Horizonte

*SELECT status\_pagamento FROM Pessoa a JOIN Participa b on a.CPF = b.CPF JOIN Obra c ON b.Id\_Obra = c.Id\_Obra JOIN Cliente d ON d.CPF = a.CPF WHERE c.Localizacao = 'Belo Horizonte'*

*SELECT status\_pagamento FROM Cliente a JOIN Participa b on a.CPF = b.CPF JOIN Obra c ON b.Id\_Obra = c.Id\_Obra WHERE c.Localizacao = 'Belo Horizonte'*

7 – Selecionar os nomes dos funcionários com salário maior ou igual a 3000 reais que não são responsáveis por nenhum documento. Apresentar os resultados em ordem alfabética.

*SELECT nome FROM Funcionario a JOIN Pessoa b on a.cpf = b.cpf LEFT JOIN Documento c ON a.cpf = c.ID\_Responsavel WHERE salario >= 3000 AND c.ID\_Responsavel is null*

*SELECT nome FROM Pessoa WHERE cpf IN (Select cpf FROM Funcionario WHERE cpf NOT IN (SELECT ID\_Responsavel FROM Documento) AND salario >= 3000) ORDER BY nome*

8 – Selecionar os tipos, localizações e status das obras em que trabalha o funcionário com maior salário.

Select tipo, localizacao, status\_obra FROM Obra a JOIN Participa b ON a.ID\_Obra = b.ID\_Obra JOIN Funcionario c ON b.cpf = c.cpf WHERE c.salario in (Select Max(salario) from Funcionario)

Select tipo, localizacao, status\_obra FROM Obra WHERE ID\_Obra IN (SELECT ID\_Obra FROM Participa WHERE CPF IN (SELECT CPF from Funcionario WHERE salario IN (Select Max(salario) from Funcionario)))

Funções de agregação (2)

9 – Selecionar o CPF, nome, o salário do funcionário e a quantidade de obras do responsável pela maior quantidade de obras

*SELECT a.cpf, nome, salario, qtd FROM ( SELECT ID\_Responsavel as cpf, count(ID\_Documento) as qtd FROM Documento GROUP BY cpf) a JOIN Pessoa b on a.cpf = b.cpf JOIN Funcionario c ON b.cpf = c.cpf WHERE qtd IN (Select max(qtd) FROM (SELECT count(ID\_Documento) as qtd FROM Documento GROUP BY ID\_Responsavel) as sq1)*

*SELECT b.cpf, nome, salario, count(Id\_Responsavel) as qtd FROM Documento a JOIN Pessoa b ON a.ID\_Responsavel = b.cpf JOIN Funcionario c ON b.cpf = c.cpf GROUP BY b.cpf ORDER BY qtd DESC LIMIT 1*

10 – Selecionar, para cada tipo de obra, a quantidade de obras registradas no sistema com ao menos um contrato de fornecimento que vence até 2022

SELECT tipo, count(\*) FROM Obra a JOIN (Select ID\_Obra FROM Fornece WHERE expiracao\_contrato < '2021-12-31' GROUP BY ID\_Obra) b ON a.ID\_Obra = b.ID\_Obra GROUP BY tipo

SELECT tipo, count(\*) FROM Obra a JOIN (SELECT DISTINCT ID\_Obra FROM Fornece WHERE expiracao\_contrato < '2021-12-31') b ON a.ID\_Obra = b.ID\_Obra GROUP BY tipo

Resultados

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| --- | Execução 1 (segundos – Tempo do servidor) | ..... 2 | 3 | 4 | 5 | Média |
| 1.1 | 0.00068288 | 0.0006012 | 0.0005774 | 0.0006498 | 0.0007778 |  |
| 1.2 | 0.0176227 | 0.0018895 | 0.0018434 | 0.0017131 | 0.0021922 |  |
| 2.1 | 0.0007796 | 0.000132 | 0.0006519 | 0.0006158 | 0.0007365 |  |
| 2.2 | 0.0017933 | 0.0013656 | 0.000877 | 0.000904 | 0.000949 |  |
| 3.1 | 0.0007962 | 0.0003109 | 0.000302 | 0.0003214 | 0.0003305 |  |
| 3.2 | 0.0004878 | 0.000444 | 0.0003963 | 0.0003088 | 0.0003256 |  |
| 4.1 | 0.0023778 | 0.0017034 | 0.0017168 | 0.0016554 | 0.0020087 |  |
| 4.2 | 0.0058893 | 0.0060674 | 0.0060329 | 0.0062795 | 0.0060817 |  |
| 5.1 | 0.0010104 | 0.0007729 | 0.0007576 | 0.0006892 | 0.000734 |  |
| 5.2 | 0.000302 | 0.0006332 | 0.0006213 | 0.00065 | 0.000345 |  |
| 6.1 | 0.0045673 | 0.0051104 | 0.0046522 | 0.004867 | 0.0055874 |  |
| 6.2 | 0.0043394 | 0.0041033 | 0.0040782 | 0.0045382 | 0.0043258 |  |
| 7.1 | 0.0077429 | 0.0083491 | 0.0080845 | 0.0081533 | 0.0087643 |  |
| 7.2 | 0.0173371 | 0.0174532 | 0.0170177 | 0.0175433 | 0.0173809 |  |
| 8.1 | 0.0545277 | 0.0549069 | 0.0585 | 0.0569878 | 0.0542288 |  |
| 8.2 | 0.0553473 | 0.0550052 | 0.0581211 | 0.0549364 | 0.055188 |  |
| 9.1 | 0.0026365 | 0.0014487 | 0.0017956 | 0.0017331 | 0.0016155 |  |
| 9.2 | 0.0069726 | 0.0041398 | 0.0033299 | 0.0038192 | 0.0042599 |  |
| 10.1 | 0.0017071 | 0.0017625 | 0.0017487 | 0.0017125 | 0.0019514 |  |
| 10.2 | 0.0019167 | 0.0019344 | 0.0021114 | 0.0018866 | 0.0018347 |  |