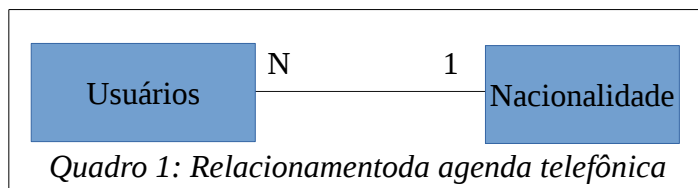


INSTITUTO FEDERAL DE CIÊNCIA E TECNOLOGIA DO RIO G. DO NORTE – IFRN
DISCIPLINA DE BANCOS DE DADOS
RESUMO DE AULA – CONCEITOS INICIAIS

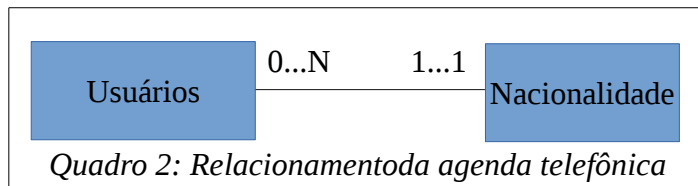
1. Cardinalidades Mínima e Máxima

No Quadro 1, presente também no resumo de aula anterior, notamos a presença do número 1 e N, junto ao relacionamento. A esses números chamamos de cardinalidade do relacionamento. Geralmente esses valores serão o número 1 e N, mas podendo também ser qualquer outro valor. Estes valores definidos no Quadro 1 são chamados de cardinalidade máxima, mas além disso define-se, de cada lado do relacionamento, o valor da cardinalidade mínima.



Por exemplo, “0 ou N” o que representaremos por “0...N” significa a presença de no mínimo zero e no máximo N em uma cardinalidade. De forma similar, “1 ou N” representaremos por “1...N” e identificam, que a tabela precisa estar associada a no mínimo um SEMPRE PARA TODOS OS REGISTROS enquanto “0...N” poderia não existir obrigatoriamente uma relação para todo o conjunto de dados.

Note que está sendo associado mais informação, assim, de cada lado do relacionamento agora deve-se ter duas informações de cardinalidades, o mínimo e máximo. Podemos reescrever o relacionamento acima como descrito no Quadro 2, após complementar as cardinalidades.



2. Relacionamentos

Existe alguns tipos de relacionamentos, inicialmente detalharemos dois deles. O relacionamento de “1 para N” representado como $1 \rightarrow N$ e o relacionamento de “N para N” representado como $N \rightarrow N$.

Obs.: O uso da setinha na representação é puramente didático e não tem nada a ver com direção do relacionamento, podendo ocorrer para qualquer dos lados sem que isso tenha nenhum significado emotivo, filosófico, conceitual ou qualquer outro.

2.1 Relacionamento $1 \rightarrow N$

Acima o relacionamento presente entre as tabelas no nosso exemplo, foi o relacionamento de 1 para N. Uma coisa importante a notar aqui é que cada relacionamento desse tipo, dá origem necessariamente a uma chave primária e uma chave estrangeira. Devido a característica de a chave estrangeira poder repetir e a primária, não, temos como regra básica que **a chave estrangeira SEMPRE estará no lado N**, da relação.

Isso ocorre pois a chave estrangeira é uma cópia da chave primária e como a chave estrangeira pode repetir, faz com que a tabela onde ela se encontra possua várias cópias de uma mesma chave primária, assim, uma chave que de um lado é primária e não pode se repetir faz com que do lado onde ela é primária, sua cardinalidade em relacionamento seja no máximo UM,

enquanto do lado onde a mesma chave é estrangeira, pode repetir, dando origem a vários registros com essa cópia.

2.2 Relacionamento N → N

Usando o mesmo exemplo, na vida real, podemos ter um usuário que tem dupla ou até mais nacionalidades e ao mesmo tempo em uma nacionalidade continua a ter vários usuários. Gerando assim, um relacionamento de N para N.

Para um relacionamento de N para N, **uma única chave estrangeira não basta**, pois **cada chave estrangeira gerará um relacionamento de 1 para N**. Precisaremos, então, de duas chaves estrangeiras, porém se colocarmos uma chave estrangeira de cada lado da relação, isto é, se no nosso exemplo colocarmos uma chave estrangeira na tabela usuários e outra na tabela nacionalidade, ao invés de gerar um relacionamento de N para N, gerará DOIS relacionamentos de 1 para N, o que causará um laço entre as tabelas, problema de integridade nos dados e não resolverá o problema, conforme explicado em sala.

A solução é bastante simples. Como não se pode colocar só uma chave estrangeira, nem se pode colocar duas chaves estrangeiras, uma em cada tabela devido a inconsistência de dados, a **solução é colocar, ambas as chaves estrangeiras em uma nova tabela**.

Assim, criaremos uma nova tabela com duas chaves estrangeiras, uma para cada tabela relacionada de N para N. A essa nova tabela chamaremos de **tabela auxiliar**. E se antes tínhamos um relacionamento de N para N, agora, teremos então um relacionamento da tabela auxiliar para cada uma das tabelas que tinham cardinalidade N, pois para cada uma das tabelas existe uma chave estrangeira na tabela auxiliar, sendo então dois relacionamentos de 1 para N.

Isso deverá ser feito sempre que tivermos um relacionamento de N para N e a essa artimanha chamaremos de **decomposição de relacionamento**. Veja no quadro 3, como fica nosso diagrama completo, com duas tabelas relacionadas de N para N após a decomposição de relacionamento.

Usuários

Nome	Telefone	pk
João	(00) 0000-0000	1
Maria	(00) 0000-0000	2
José	(00) 0000-0000	3
Joaquim	(00) 0000-0000	4

Nacionalidade

Rua	pk
Brasileiro	A
Espanhol	B

Auxiliar

fk_usuarios	fk_enderecos
1	A
2	A
3	A
1	B

1

N

1

N

Quadro 3: Agenda Telefônica com relacionamento N para N já decomposto

Pensando apenas no usuário João cuja chave primária é 1, podemos olhar na tabela auxiliar quais registros existem onde a chave estrangeira para usuario é 1, isso nos dará a nacionalidade A e B (dois registros), que verificando na tabela nacionalidade, corresponde a Brasileiro e Espanhol, respectivamente. Com isso provamos que nesse tipo de ligação, um usuário pode ter várias nacionalidades.

De forma análoga, se pensarmos apenas em uma nacionalidade, por exemplo, Brasileiro, cuja chave primária é a letra A, podemos verificar na tabela auxiliar quais registros existem onde a chave estrangeira para nacionalidade é a letra A, o que nos dará os usuários 1, 2 e 3, que correspondem a João, Maria e José, respectivamente, comprovando que um endereço, nesse tipo de ligação, pode ter vários usuários.

3. Banco de Dados Avançado

Note que no nosso exemplo anterior, na tabela usuários podia ter usuários com nomes repetidos e isso realmente pode ocorrer na vida real, ou seja, dois usuários com mesmo nome. Porém se tivéssemos no banco de dados uma coluna com o cpf, por exemplo, esse valor não pode repetir em nosso banco de dados, pois significaria usuário repetido causando confusão na hora de manter esses dados atualizados, identificar o login ou e-mail correto, por exemplo. Para solucionar esses problemas, devemos nos utilizar da chave de unicidade.

3.1 Chave de Unicidade

As chaves primárias e estrangeiras já foram explicadas anteriormente, porém temos ainda mais um tipo de chave. A chave de unicidade serve como o nome diz, apenas para evitar que UMA determinada coluna se repita. Muito útil por exemplo ao colocar o cpf do usuário no banco de dados, evitar que ele se repita, assim, ao inserir um novo usuário com o mesmo número de cpf, o banco de dados irá proibir esta inserção.

Note que não existe nenhuma relação direta entre chave de unicidade e chave primária. São tipos distintos de chave e em nenhuma hipótese produzem o mesmo efeito. Isto é assunto para uma outra aula.

3.2 Chave estrangeira como cópia da chave primária

Notamos que a chave primária é obrigatória quando temos de usar uma chave estrangeira, pois esta precisará copiar o valor da chave primária. Isto não é totalmente verdade. A chave estrangeira pode copiar qualquer valor que seja único de outra tabela.

Na modelagem de bancos de dados, entretanto, adotamos como padrão a cópia da chave primária, pois além de ser única, a chave primária também é não nula, o que garante a relação entre quaisquer registros, pois caso fizessemos a cópia de uma informação vazia (uma chave única que pode ser nula), não seria possível pois não haveria o que copiar.

Imagine por exemplo, uma chave estrangeira que copie o valor do cpf do usuário. Caso o usuário seja estrangeiro, criança ou não registrado, ele não terá cpf, então não será possível usar essa chave estrangeira e consequentemente o sistema que usa esse banco de dados não funcionará.

Dessa forma, devemos usar a chave estrangeira sempre efetuando a cópia de uma chave primária, o que nos poupará de diversos problemas.

3.3 Obrigatoriedade da chave primária

A chave primária, seguindo apenas o conteúdo acima, só seria obrigatória caso exista uma chave estrangeira que deverá copiar esse conteúdo. Porém devemos colocar a chave primária em todas as tabelas, isto porquê garantirá que o banco de dados funcionará mais rápido servindo como critério de indexação, além de nos garantir que dois registros são diferentes, por exemplo, caso dois usuários tenham mesmo nome, e todos os demais dados iguais, as chaves primárias serão diferentes, nos permitindo diferenciá-los.

3.4 Chaves Compostas (Formadas por mais de um atributo)

Vimos até agora que as chaves são colocadas em atributos da tabela. O que não vimos ainda é que estas chaves podem ser criadas a partir da união entre atributos. Imagine por exemplo uma tabela com dados de produtos com os atributos: Nome do produto, Volume e Lote.

Apesar que qualquer desses atributos podem vir a se repetir no banco de dados, não desejamos que eles se repitam todos ao mesmo tempo, isto é, um cadastro repetido de produtos de mesmo nome, volume e lote. Dependendo do produto, algum desses campos eventualmente pode ser vazio, porém não devem ser todos vazios ao mesmo tempo.

Caso digamos que a união destes atributos forma uma única chave primária, o que estamos dizendo não é que nenhum dos atributos pode ser vazio nem tampouco que todos os atributos são únicos, como em uma chave primária tradicional.

O que vale nas chaves compostas, isto é, chaves formadas por um conjunto de atributos é sobre a sua combinação. Neste caso a combinação dos três atributos não poderia ser nula e precisa ser única, eliminando a possibilidade de produto repetido e por a combinação ser necessariamente não nula, significa que ao menos um dos atributos precisa estar preenchido, o que pode ser combinado com mais regras para garantir que mais atributos devem ser não nulos ou únicos individualmente caso necessário.

3.5 Chaves Mistas (Primárias e estrangeiras ao mesmo tempo)

Veja por exemplo no Quadro 3, que poderíamos ter, em todas as linhas informando que o mesmo usuário tem a mesma nacionalidade, repetidas vezes. Isso não seria contrário a nenhum dos conceitos vistos, porém seria um problema por apresentar dados repetidos.

Vimos também que a garantia de uma informação não repetir são as chaves primárias. Vimos também que uma chave pode ser composta, isto é, formada por mais de um atributo. O que não vimos é que a chave primária pode, ao mesmo tempo, ser estrangeira.

Por exemplo, se no Quadro 3, na tabela Auxiliar, a combinação das duas chaves estrangeiras formar uma única chave primária, então poderemos repetir a chave estrangeira para usuário, poderemos repetir também a chave estrangeira para nacionalidade, porém a combinação das duas, não poderia ser repetida. Isto é, para uma mesma chave estrangeira de usuário, as nacionalidades precisam ser diferentes e vice-versa.

A esse tipo de chave, primária e estrangeira ao mesmo tempo, usaremos a sigla PF (do inglês, Primary and Foreign Key). Veja que cada tabela só pode ter uma chave primária, porém essa única chave primária pode ser a composição de mais de uma coluna.

Note que caso, tornemos uma única chave estrangeira em chave mista, isto é, primária e estrangeira (ou única e estrangeira), o relacionamento criado por esta chave não será mais 1->N e sim um relacionamento de 1 → 1. Isto porquê já que de ambos os lados do relacionamento, os rótulos usados para identificar a relação, serão chaves primárias, isto é, não poderão repetir.

Note também que em uma tabela auxiliar, quando unimos duas colunas para formar uma chave primária, as colunas podem repetir suas informações, desde que não repitam a combinação de ambos ao mesmo tempo, assim, não é afetado o relacionamento com as demais tabelas.

3.6 Atributos Não Nulos

Ao dizer que uma chave estrangeira foi criada, nada a impede de ser vazia, assim, influencia na cardinalidade pois para um registro, poderá não se relacionar a ninguém gerando cardinalidade mínima e máxima de “0...1” do outro lado do relacionamento, isto porque caso esteja vazia, se relaciona a zero elementos do outro lado do relacionamento e caso preenchida se relaciona a um elemento.

Cada atributo, individualmente, podemos dizer que deve ser não nulo, isto é, deve obrigatoriamente estar sempre preenchido. **Caso digamos que uma chave estrangeira é não nula, afetará então a cardinalidade mínima do relacionamento pois como sempre estará preenchida, então sempre estará relacionada a no mínimo um.**

LISTA DE EXERCÍCIOS - \ô/

O modelo físico de dados é quando temos todas as informações necessárias para implementação do diagrama, isto é, todas as chaves primárias, estrangeiras, unicidade, atributos não nulos, chaves mistas, tabelas auxiliares e seus relacionamentos detalhados com suas cardinalidades mínima e máxima.

1. Mostre um diagrama físico envolvendo duas tabelas usando os relacionamentos abaixo. Por simplicidade, pode-se nomear as tabelas de A e B.

- a) “0...1” → “0...1”
- b) “1...1” → “0...1”
- c) “0...1” → “0...N”
- d) “1...1” → “0...N”
- e) “0...2” → “0...1”
- f) “1...2” → “0...1”
- g) “2...2” → “0...1”
- h) “0...2” → “0...N”
- i) “1...2” → “0...N”
- j) “2...2” → “0...N”
- k) “0...N” → “0...N”

2. Defina e mostre um exemplo onde seja necessário o uso de:

- a) chave composta
- b) chave mista
- c) chave de unicidade
- d) atributo não nulo