

AULA 02: Modelo Relacional e SQL

SUMÁRIO	PÁGINA
1. Introdução	1
2. Modelo Relacional	3
2.1. Conceitos	3
2.2. Chaves	8
2.3. Mapeamento do MER para o Modelo Relacional	18
2.4. Restrições de Integridade	23
2.5. Álgebra Relacional	29
2.6. Linguagens de Banco de Dados	34
3. Questões Comentadas	35
4. Resumo	39
5. Lista das Questões Apresentadas	43
6. Gabaritos	51

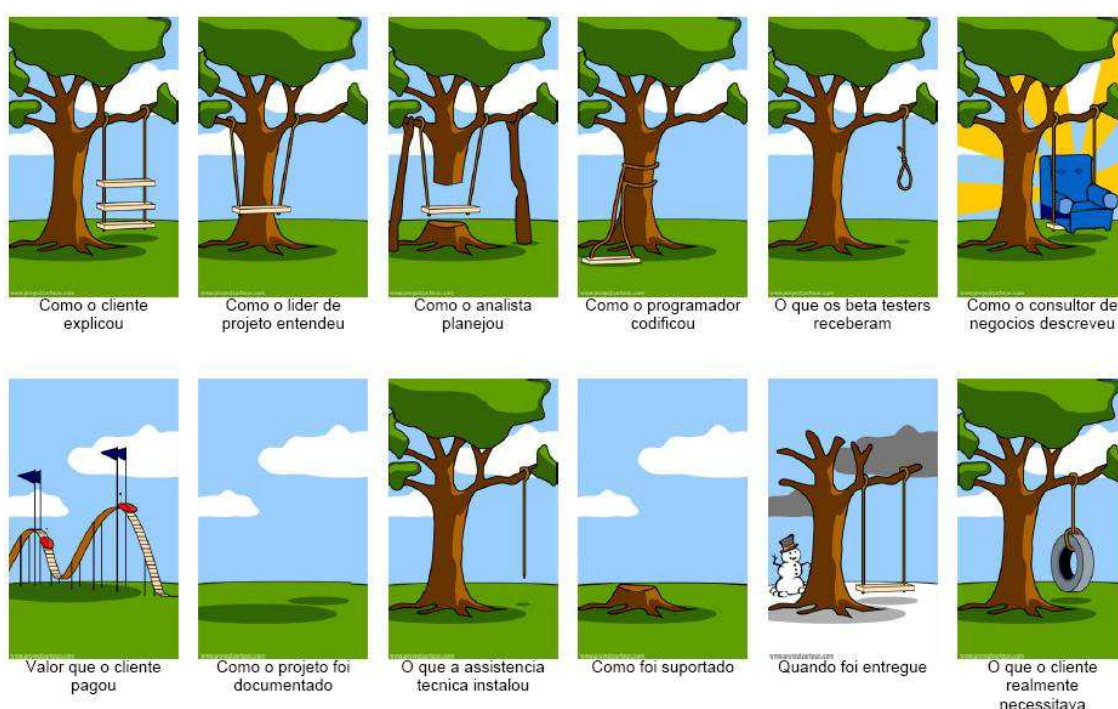
1. INTRODUÇÃO

Saudações caros(as) amigos(as),

Hoje vamos à nossa terceira aula de **Conhecimentos de Banco de Dados**, tratando do tema Modelo Relacional. Como vocês devem lembrar, ele é o nosso modelo lógico, e ao seu término estamos mais perto de montar nosso Banco de Dados propriamente dito, faltando somente o modelo físico (ai já entra na parte do SQL).

Em termos de projeto de Banco de Dados, a fase mais importante já passou, que foi a construção do modelo conceitual (no nosso caso, com o uso do MER). Foi naquela fase que entendemos o problema, e vocês verão que passar do MER para o modelo relacional é uma simples questão de mapeamento do que já temos, ou seja, iremos transformar nosso MER em Modelo Relacional. Por outro lado, se na fase anterior entendermos errado as especificações do cliente, nosso erro vai se propagando para as próximas fases.

É como se o cliente desejasse uma cadeira para ele, mas nós entendêssemos que ele deseja uma mesa. Dessa forma, fizemos nosso modelo conceitual representando uma mesa, e no modelo lógico vamos refinar nosso entendimento, aperfeiçoando a mesa, especificando suas dimensões, seus materiais etc. Quando entregarmos o produto final, a mesa vai estar muito bem definida e construída, só que como eu disse, o que o cliente quer é uma cadeira. Tem uma figura famosa da Engenharia de Software que retrata isso.



Bem, isso que eu disse agora foi só para contextualizar, e com certeza não vai interessar muito para a nossa prova, pois está mais ligado a área de engenharia de software. O que eu quero que vocês tirem disso é que o Modelo Relacional não vai criar nada de novo daquilo que nós definimos no MER, pois a partir desse ponto vamos apenas transformar nosso DER, de forma que ele possa ser empregado em uma tecnologia de Banco de Dados específica, que no caso é Tecnologia dos Bancos de Dados Relacionais. Entendam essa tecnologia como a forma que o Banco de Dados vai armazenar e tratar as informações persistidas.

Antes de começarmos o assunto propriamente dito, só mais um detalhe. Vou colocar um pequeno tutorial disponível, para que vocês instalem um SGBD (MySQL) nos seus computadores pessoais. O objetivo disso é que vocês possam ver como funciona de verdade o SQL. Não é uma atividade obrigatória, e vocês poderão estudar o SQL só na teoria, pois eu sei que o tempo está curto até a prova. Mas acredito que se usarem o próximo final de semana, gastarão menos de uma hora para instalar o SGBD e configurar seu uso. Aprender SQL sem digitar os comandos e ver o resultado é meio que aprender violão sem tocar no instrumento. Mas fica como algo opcional, quem não quiser fazer pode acompanhar somente pelas aulas.

Então, vamos ao trabalho.

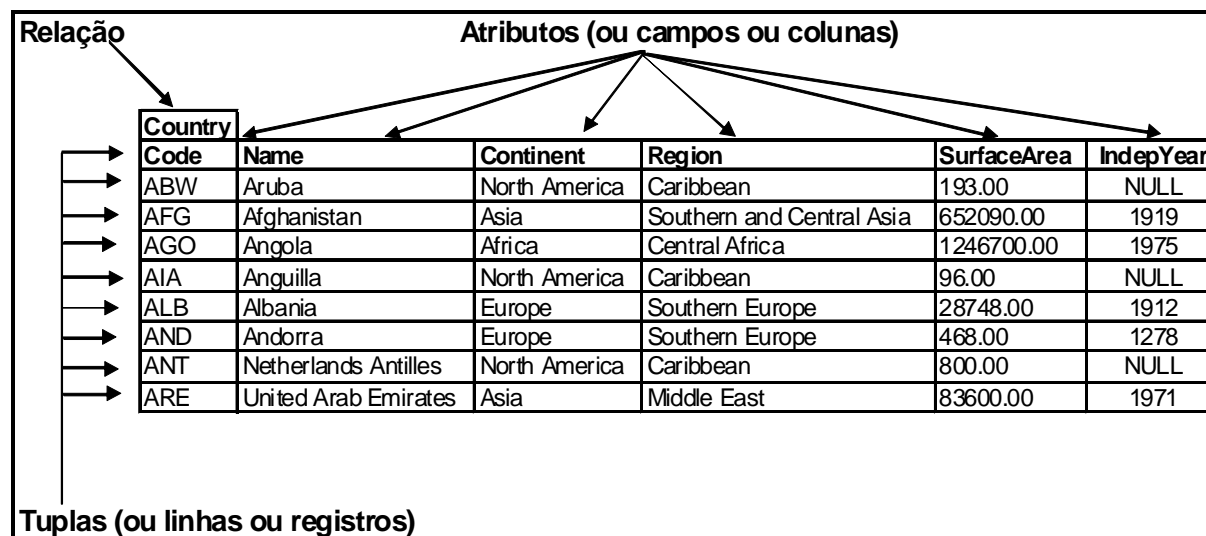
2. MODELO RELACIONAL

2.1. CONCEITOS

O Modelo relacional foi introduzido por Codd, um pesquisador da IBM, em 1970. Esse modelo usa o conceito de relação matemática (algo como uma tabela de valores) como seu bloco de construção básica. Sua base teórica é a teoria dos conjuntos e lógica de predicados. Apesar de ser um modelo matemático, não se preocupem, veremos que é algo muito simples. Todos aqueles SGBD que citamos em aulas anteriores (Oracle, MySQL, SQL Server, PostGre, DB2) utilizam o Modelo Relacional como base. Portanto, são considerados SGBD Relacionais.

O Modelo Relacional representa os dados em um Banco de Dados como um conjunto de tabelas, que no modelo são denominadas **Relações**. Cada Relação terá um nome, que será único, e um conjunto de atributos, com seus respectivos nomes e domínios. O conceito de atributo é semelhante àquele que vimos no MER. E o que é o domínio? Também vimos na aula passada. **O domínio de um atributo é o conjunto de possíveis valores para o mesmo.**

Com relação a atributos, eles são o que comumente denominamos campos, ou colunas da **Relação**. Cada linha de uma **Relação** é denominada uma **tupla** (o que mais a frente entenderemos como registro). Parece um pouco complicado? Na verdade não é, vamos a uma figura, que fica mais fácil.



Na figura acima temos uma **Relação** chamada de Country. Essa **Relação** é composta por **atributos** (campos) que descrevem cada elemento da **Relação**. Os atributos são Code, Name, Continent, Region, SurfaceArea e IndepYear. Cada linha da **Relação** tem valores diferentes para os seus atributos, representando diferentes elementos desse conjunto. Para cada linha damos o nome de **tupla**. Uma **tupla** é equivalente a um registro de uma tabela, pois representa um elemento da mesma. No Modelo Relacional utilizamos a terminologia **Relação**, **atributo** e **tupla**. Mas veremos que esses elementos correspondem no modelo físico a tabela, campo e registro, respectivamente. Quis colocar as duas denominações para que possamos associar os conceitos, mas de agora em diante, como ainda estamos no Modelo Relacional, vamos falar somente em **Relação**, **atributo** e **tupla**.

Uma maneira comum de se representar uma Relação é através do seu **esquema de relação**. Um esquema de relação nada mais é do que

uma descrição dos atributos que compõe a Relação. Esses atributos normalmente são seguidos pelo seu domínio, mas às vezes encontramos somente o nome dos atributos. Assim, o esquema da relação Country seria representado da seguinte forma:

Country (Code, Name, Continent, Region, SurfaceArea, IndepYear).

Ou então, pode ser representado com os domínios dos atributos:

Country (Code:String, Name:String, Continent:Enumeração, Region:String, SurfaceArea:Real, IndepYear:Inteiro).

Na primeira representação colocamos apenas o nome da Relação, seguida pelo nome dos atributos entre parênteses, e separados por vírgula. Na segunda representação, acrescentamos o domínio dos atributos, colocando após o nome do atributo e separando deste por dois pontos. Não se preocupem agora com esses domínios que eu coloquei, vai chegar o momento de falarmos em tipos de dados suportados pelos SGBD. Só por curiosidade, String é um domínio que significa um valor literal (qualquer sequência de caracteres), Enumeração significa que teremos nesse campo valores pré-determinados (como Ásia, África etc), Real significa que aceitará valores numéricos com casas decimais (ou seja, do conjunto dos números reais) e Inteiro que aceitará valores do conjunto dos inteiros (afinal, não existe o ano de 2007,5, existe o ano de 2007).

As Relações têm um grau. **O grau da Relação é simplesmente o número de atributos que a mesma possui.** Assim, a Relação Country tem grau 6.

O **instante (snapshot) da Relação** é a fotografia de um determinado momento da Relação. Por exemplo, na Relação Country mostrada, temos um instante da Relação, porque no momento representado na figura ela tinha somente aquelas tuplas. Mas como as

Relações são dinâmicas, se tirarmos o instante da mesma Relação amanhã, pode ser que os valores sejam diferentes (venham mais ou menos tuplas, ou mesmo os valores contidos nos atributos sejam diferentes destes apresentados).

As Relações têm algumas características importantes, a seguir descritas:

- **Ordenação:** a ordem das tuplas e dos atributos não importam para a Relação. Isso quer dizer que, como as tuplas são matematicamente definidas como elementos de um conjunto (a Relação), a ordem delas em nada altera a Relação. Isso vale também para os atributos. Ou seja, se temos duas Relações com os mesmo atributos, mas em ordens diferentes, estamos tratando da mesma relação.
- **Atomicidade:** todos os atributos devem ter valores atômicos, ou seja, únicos. Assim, não é possível ter atributos multivalorados ou compostos no Modelo Relacional, ao contrário do MER, que aceitava atributos multivalorados e atributos compostos.
- **Identidade:** cada atributo de uma Relação tem um nome que é único para aquela Relação. Vimos que a Relação Country tem 6 atributos, e cada um tem um nome diferente. Não poderiam existir dois atributos com o mesmo nome dentro da Relação. Contudo, duas Relações distintas podem compartilhar o mesmo nome para algum atributo. Por exemplo, poderemos ter uma Relação Aluno, com o atributo nome, e no mesmo minimundo ter uma Relação Professor, com o atributo nome.
- **Unicidade:** todas as tuplas devem ser únicas. Assim, no Modelo Relacional não podem haver duas tuplas com todos os valores exatamente iguais. Por exemplo, não faz sentido na

Relação Country ter duas tuplas com os valores Code=ABW, Name=Aruba ... IndepYear=NULL. Cada tupla deve ter no mínimo um valor diferente.

É meus amigos, imagino que esteja um pouco chato ver tantos conceitos, mas eles são necessários. As coisas ficam mais claras no momento que nós começarmos a aplicar o modelo. Vamos então a umas questões para tirar um pouco do estresse.

1. (ESAF/Analista Desenvolvimento de Sistemas/ANA 2009)
O modelo de dados baseado numa coleção de tabelas que representam dados e as relações entre eles é denominado modelo

- a) relacional.**
- b) entidade/relacionamento.**
- c) baseado em objetos.**
- d) de dados semiestruturados.**
- e) objeto/relacionamento.**

Comentários:

Dos modelos que comentamos, aparecem nas alternativas o relacional e o de entidades/relacionamentos. Falei para vocês que as Relações no Modelo Relacional são as tabelas. Mesma não estando 100% correto se falar em tabelas no Modelo Relacional, pois o mais correto é Relação, é comum usar tabela como sinônimo de Relação. Assim, temos que o modelo pedido na questão é o Modelo Relacional.

Gabarito: Letra a.

2. (ESAF/AFRF/SRF 2002) Em um banco de dados relacional, os objetos que realmente armazenam os dados são

- a) as chaves primárias.**
- b) os relacionamentos.**

c) as tabelas.

d) as transações.

e) os procedimentos armazenados.

Comentários:

Essa ficou fácil, não é? Conforme já comentamos, são as tabelas que armazenam os dados

Gabarito: Letra c.

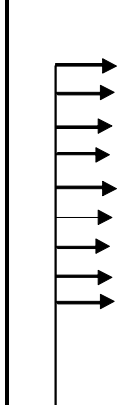
2.2. CHAVES



Chegou o momento de tratar de uma questão importante, comentada por alto na última aula, que são as chaves. Bem, já sabemos que duas tuplas distintas de uma Relação não podem ter exatamente os mesmo valores em cada um de seus atributos, ou seja, elas não podem ser iguais.

Geralmente dentro de uma Relação existem diversos subconjuntos de atributos de um esquema de relação com a propriedade de que, duas tuplas, em qualquer estado (ou instante) dessa Relação, não tenham as mesmas combinações de valores para os atributos do subconjunto. Esse conjunto de atributos com essa característica é denominado chave. Assim, **podemos definir uma chave como o conjunto de um ou mais atributos que determinam a unicidade de cada tupla (registro).** Vamos ver que chave é um gênero que comporta muitas espécies diferentes.

Para uma Relação, temos diversos subconjuntos de atributos que podem servir como chave. Cada conjunto distinto de atributos, que pode identificar de forma unívoca uma relação, é chamado de superchave. Vamos ver outra relação para nos basearmos.

Relação		Atributos (ou campos ou colunas)					
		CPF	Nome	Nascimento	Endereco	Cidade	Estado
		111222	Alberto	01/01/1975	Rua 17 n 76	Manaus	AM
		123445	Ana	03/05/1947	Rua A quadra 1	Brasília	DF
		345321	Luiza	12/10/1990	Quadra L casa 6	Salvador	BA
		237980	Francisco	17/02/1980	Cond Maresia ap 301	Fortaleza	CE
		567987	Guilherme	25/08/1970	Rua 101, n 303	Maceió	AL
		563029	Patrícia	15/12/1989	Av das Torres n 130	São Paulo	SP
		908123	Milena	28/11/1974	Quadra H ap 304	Boa Vista	RR
		985209	Andrea	20/05/1946	Rua das Lamentações, 120	Rio Branco	AC
		Tuplas (ou linhas ou registros)					

Partindo para a análise da figura, temos uma Relação chamada Cliente. O esquema dessa Relação é (sem domínio):

Cliente (CPF, Nome, Nascimento, Endereço, Cidade, Estado).

Só repetindo o conceito, **superchave** é um conjunto de um ou mais atributos que, tomados coletivamente, nos permitem identificar de maneira unívoca uma entidade em um conjunto de entidades. Em outras palavras, não podem existir duas ou mais linhas da tabela com o(s) mesmo(s) valor(es) de uma superchave. Ai vocês me perguntam, mas esse não é conceito de chave? Sim, o conceito é o mesmo, mas veremos que a superchave (espécie) tem uma característica que nem toda chave (gênero) tem. Ou seja, toda superchave é uma chave, mas o contrário não é verdade. Agora vamos encontrar as superchaves da nossa Relação.

Vocês concordam que a reunião de todos os atributos de uma Relação forma uma superchave? Não falamos que em uma Relação não

pode haver duas tuplas exatamente iguais? Sim, falamos. Então, se não temos duas tuplas exatamente iguais, a reunião de todos os atributos formam uma superchave, pois é um conjunto de atributos que tomados coletivamente permitem identificar de forma única uma tupla. Vamos representar as chaves com a notação $S_n = \{\text{Atributos}\}$. Dessa forma, a primeira superchave encontrada, que eu vou chamar de S1, vai ficar assim:

$S1 = \{\text{CPF, Nome, Nascimento, Endereço, Cidade, Estado}\}$

Agora vamos continuar. Suponha que eu queria formar uma chave com os todos os campos da Relação Cliente, menos o campo Estado. A chave que eu vou chamar de S2 ficará da seguinte forma:

$S2 = \{\text{CPF, Nome, Nascimento, Endereço, Cidade}\}$

A pergunta é, S2 é uma superchave? Podem existir duas tuplas com os valores dos atributos de S2 iguais? Não, não pode, pois não podemos ter duas tuplas com o mesmo CPF, Nome, Nascimento, Endereço e Cidade iguais, só mudando o atributo Estado. Então S2 é uma superchave.

Mais um exemplo, agora pegando o seguinte subconjunto de atributos: CPF, Cidade. Vou chamar essa chave de S3, que ficará da seguinte forma:

$S3 = \{\text{CPF, Cidade}\}$

De novo a pergunta: Posso ter duas tuplas distintas (que representam dois clientes) com os mesmo valores de CPF e Cidade? Não posso, cada tupla vai ter essa combinação de valores diferentes. Então, temos mais uma superchave.

Por fim, vou pegar o subconjunto Cidade e Estado para formar uma chave, que chamarei de S4. Ficar da seguinte forma:

$S4 = \{\text{Cidade, Estado}\}$

Minha pergunta é, podem existir duas tuplas com os mesmos valores para os campos Cidade e Estado? Sim, claro que pode, eu posso ter diversos clientes na da mesma cidade e estado. Então, **S4 não é uma chave, muito menos uma superchave.**

E um campo sozinho, é uma superchave? Vejamos, imaginem agora que tenho o subconjunto somente com o campo CPF. Formando a chave S5, temos:

$$\mathbf{S5 = \{CPF\}}$$

Ora, duas tuplas (que representam dois clientes distintos) podem ter o mesmo valor para CPF? Claro que não, então **S5 é uma superchave.**

O que quis mostrar a vocês é que dado um conjunto de atributos, existem vários subconjuntos deles que têm a característica de identificar de forma única uma tupla, tornando-se superchave.

Pois bem, agora que já sabemos o que são superchaves, vamos a outro conceito **muito importante**, que são **chaves candidatas**. São superchaves de **tamanho mínimo**, candidatas a serem chaves primárias da Relação. Ou seja, são um atributo ou conjunto de atributos que permitem identificar de forma inequívoca qualquer tupla dessa relação, **mas que não pode ser reduzido sem perder qualidade.**

Então vamos analisar nossas chaves já definidas, que sabemos que são superchaves, com exceção de S4, que não é nem chave nem superchave.

S1 é uma chave candidata? Pensem o seguinte, existe pelo menos um atributo que eu posso retirar de S1, e ainda assim o subconjunto resultante ainda ser superchave? Sim, existem vários atributos que posso retirar de S1 e ainda assim obter uma superchave. **Então S1 não é uma chave candidata.**

Agora S2, partindo do mesmo raciocínio. Se eu retirar de S2, por exemplo, o atributo Nascimento, os atributos restantes (CPF, Nome, Endereço e Cidade) continuam formando uma superchave? Sim, então **S2 também não é uma chave candidata.**

E S3? Bem, temos só os atributos CPF e Cidade. Existe algum atributo que eu possa retirar, e mesmo assim continuar tendo uma superchave? Sim, se eu retirar o atributo Cidade, o atributo CPF sozinho continua sendo uma superchave. Então, **S3 também não é uma chave candidata.**

E S5? Bem só temos um atributo. Assim, não tem nem o que tirar. Então, **S5 é uma chave candidata.** Como vocês devem ter inferido, **toda superchave que tem apenas um atributo é uma chave candidata.** Mas não se restrinjam a esse exemplo. **Nem toda chave candidata tem apenas um atributo.** Querem um exemplo?

Imaginem uma relação que guarde a participação de funcionários em projetos de uma empresa. Essa Relação teria, entre outros atributos, os seguintes: **NumeroProjeto, MatriculaFuncionario, DataInicio, CargaHoraria.** Vejam que eu represento atributos como palavras únicas, sem espaços entre elas, e é assim que representamos no Modelo Relacional e daqui para frente. Bem, não vou analisar todas as possibilidades de superchave, mas pergunto para vocês (valendo um vale transporte): Existe algum atributo que, individualmente, pode ser superchave? NumeroProjeto não pode, pois para determinado projeto existirão diversos funcionários. MatriculaFuncionario também não pode, pois esse valor se repete em diversas tuplas, uma vez que um funcionário pode participar de diversos projetos. DataInicio e CargaHoraria também não podem individualmente identificar uma tupla. Então, das combinações possíveis, percebemos que existe apenas uma chave candidata, que é formada por NumeroProjeto e MatriculaFuncionario.

S6 = {NumeroProjeto, MatriculaFuncionario}

Podem procurar, se alguém achar outra chave candidata nessa Relação, além do vale transporte vai ganhar um pastel de vento!!

Bem, por fim, devemos tratar agora das chaves primárias. Entre as chaves candidatas possíveis, devemos escolher para cada relação uma, e somente uma, chave primária. **Chave primária nada mais é um conjunto de um ou mais campos, cujos valores, considerando a combinação de valores de todos os campos da tupla, nunca se repetem e que podem ser usadas como um índice para os demais atributos da relação, escolhida entre as chaves candidatas.** Em chaves primárias, **não pode haver valores nulos** nem repetição de tuplas. Isso é muito importante e cobrado em prova, e vocês não podem esquecer. **Nenhum atributo de uma chave primária pode conter o valor Nulo.** Mas professor, se eu tiver uma chave primária que é a combinação de 10 campos (atributos), nenhum pode ser nulo? Não, nenhum. Essa propriedade é normalmente conhecida como **restrição de integridade de entidade.**

Ufa!!! Agora que vocês sabem quase tudo que precisam saber sobre chaves, vamos a algumas questões. Ainda falta um tipo de chave, mas veremos depois.

3. (ESAF/Analista Desenvolvimento Sistemas/ANA 2009) Um conjunto de um ou mais atributos, tomados coletivamente, para identificar unicamente uma tupla numa relação, é denominado

- a) chave assimétrica.**
- b) chave simétrica.**
- c) superchave.**
- d) chave secundária.**
- e) chave de tupla.**

Comentários:

Bem, e aí? Depois desse “falatório” todo essa ficou fácil, não é? A questão está se referindo à superchave. Por sinal, vale frisar que chave de tupla é invenção do examinador, e chaves assimétrica e simétrica não tem nada a ver com Banco de Dados, são conceitos de criptografia

Gabarito: Letra c

4. (ESAF/TRF/SRF 2006) Analise as seguintes afirmações relacionadas a Bancos de Dados:

I. Em uma tabela, quando existir uma combinação de colunas que sirva para identificar todos os registros dessa tabela, essa combinação poderá ser escolhida como uma chave primária composta.

II. Em um banco de dados, quando se deseja garantir que, em uma coluna ou combinações de coluna, a qualquer momento, nenhum par de linhas da tabela deva conter o mesmo valor naquela coluna ou combinação de colunas, é necessário definir uma chave primária.

III. Uma das regras da integridade do modelo relacional é possibilitar que um atributo que participe da chave primária de uma relação básica aceite um e somente um valor nulo.

IV. Normalização é o processo de se reunir todos os dados que serão armazenados em um certo banco de dados e concentrá-los em uma única tabela.

Indique a opção que contenha todas as afirmações verdadeiras.

a) II e III

b) I e II

c) III e IV

d) I e III

e) II e IV

Comentários:

Vamos analisar cada item:

I. Sim, vimos que uma chave primária é um conjunto de atributos (aqui tratados como colunas) que podem identificar todas as tuplas da relação, ou como está na questão, todos os registros da tabela. Lembram quando eu relacionei Relação com tabela, tupla com registro e atributo com campo ou coluna? Pois é, as bancas chamam hora de um jeito, ora de outro. Por fim, a questão fala em chave primária composta. Está correto, composta aqui significa que ela tem mais de uma coluna (atributo). Se tivesse só uma, seria chave primária simples. Então, nosso item está todo correto.

II. Sim, já vimos que uma chave primária serve para garantir que em nenhum momento existam duas linhas (ou registros ou tuplas) com os mesmos valores, sendo que os próprios campos das chaves primárias não podem se repetir em mais de uma tupla. Item correto. Vejam que a ESAF fala aquilo que estudamos, mas às vezes coloca uma linguagem um pouco mais confusa. Mas nada que o nosso estudo não possa derrubar.

III. Conforme eu alertei, nenhum atributo que participa da chave primária pode ter valores nulos. Como eu já expliquei também, isso é uma regra de integridade do Banco de Dados, chamada de restrição de integridade de entidade. Item incorreto.

IV. Ainda não falamos de normalização, é um assunto que virá mais a frente. Mas acreditem em mim, o item está incorreto

Gabarito: Letra b

5. (ESAF/ACE-Sistemas/TCU 2002) Analise as seguintes afirmações relativas a Banco de Dados:

I. Uma chave primária não pode desempenhar a função de identificação única;

II. Um modelo conceitual de banco de dados representa a estrutura de dados de um Banco de Dados com os recursos e particularidades de um Sistema de Gerenciamento de Banco de Dados específico;

III. Entidade pode ser definida como um objeto que existe no mundo real, com uma identificação distinta e com significado próprio;

IV. Uma das regras da integridade do modelo relacional afirma que nenhum campo que participe da chave primária de uma tabela básica pode aceitar valores nulos.

Indique a opção que contenha todas as afirmações verdadeiras.

- a) I e II**
- b) II e III**
- c) III e IV**
- d) I e III**
- e) II e IV**

Comentários:

I. Incorreto. Uma chave primária deve desempenhar essa função.

II. Não, como vimos, o modelo conceitual é de alta abstração, não estando ligado à tecnologia e SGBD específico.

III. Apesar de um pouco restrito esse conceito de entidade, pode ser considerado correto. Lembrando que nem toda entidade existe no mundo real. Assunto da aula passada, ainda lembram? Item correto.

IV. Mais uma vez essa questão, e agora colocada de maneira correta.

Gabarito: Letra c

6. (ESAF/TTN-Informática/SRF 1998) Em relação aos bancos de dados é correto afirmar que

- a) a chave primária define uma ordem padrão para a ordenação dos campos de um registro**
- b) os campos lógicos podem armazenar strings de caracteres quaisquer**
- c) as colunas das tabelas que compõem um banco de dados são chamadas campos e as linhas são chamadas registros**
- d) a chave primária só pode ser formada por um único campo**
- e) os bancos de dados relacionais são também chamados de bancos de dados simples**

Comentários:

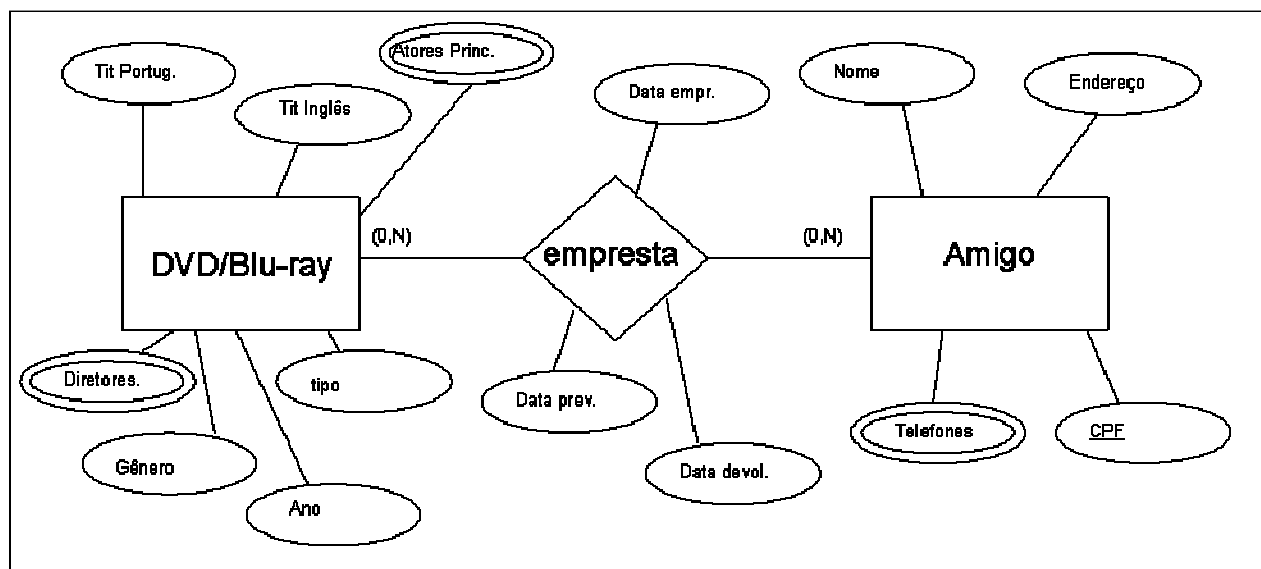
- a) A chave primária nada tem a ver com ordem de campos. Ela identifica de forma única um registro (tupla) em uma tabela (relação).
- b) Campos lógicos são um dos tipos de campo que eu prometi tratar mais a frente. Mas o seu domínio não são strings. Eles só aceitam dois tipos de valores, Verdadeiro e Falso.
- c) Definição correta, conforme já vimos. As colunas são os campos, ou atributos, e as linhas são os registros, ou tuplas.
- d) Já sabemos que a chave primária pode ser simples ou composta.
- e) Não, isso não existe.

Gabarito: Letra c

2.3. MAPEAMENTO DO MER PARA O MODELO RELACIONAL

Agora que já vimos um bocado do Modelo Relacional, podemos mapear o MER para o Modelo Relacional. Para fazer isso, basta seguir

algumas regrinhas. Não vamos ver todas as regras em detalhe, porque não devem ter questões com essas regras. Vocês só precisam entender como ligamos os dois modelos. Bem, para isso vou recuperar o DER que fizemos aula passada.



Passo 1 - Mapeamento das entidades regulares (ou simples, ou fortes): Sendo bem objetivo, cada entidade regular vai virar uma Relação, e os atributos que não multivalorados viram também atributos da Relação. Nos atributos compostos (como poderia ser o caso de endereço, mas não representei assim), cada atributo decomposto vira também um atributo da Relação. Seguindo esse passo, teremos inicialmente duas Relações, uma que eu vou chamar de DVD (só para simplificar) e outra que vou chamar de Amigo. Vamos a elas:

DVD (TituloPort, TituloIng, Tipo, Gênero, Ano)

Amigo (CPF, Nome, Endereço)

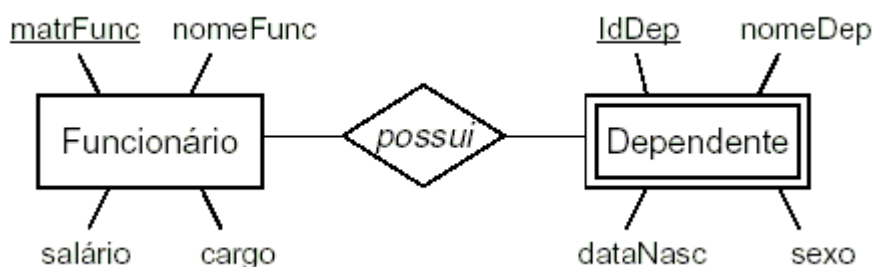
Vejam que diretores e atores principais não entram na relação DVD, pois são multivalorados. O mesmo com Telefones, em Amigo. Nesse passo, com o nosso conhecimento de chaves, devemos escolher uma

chave primária para cada Relação, se possível. Em Amigo, indico CPF como chave primária. Em DVD, não consegui encontrar de cara uma chave primária. O Título em Português e /ou em inglês podem se repetir (além do mesmo nome ser usado nos remakes, posso ter DVD e blu-ray iguais). Ademais, esses campos são passíveis de valores nulos, não podendo compor a chave. Nesses casos, uma das soluções é criar um código, ou Id (de identificador), que é um número sequencial com um valor diferente para cada tupla. É de praxe colocar os campos que compõe a chave primária na frente, e sublinhados. Então, nossas relações ficam assim:

DVD (IdDVD, TituloPort, TituloIng, Tipo, Gênero, Ano)

Amigo (CPF, Nome, Endereço)

Passo 2 - Mapeamento das entidades fracas: quase igual ao passo 1, com uma pequena diferença. Lembram que a entidade fraca só existe se existir a forte, e que todo elemento da entidade fraca deve estar relacionado a pelo menos um elemento da entidade forte? Então, a diferença é que a Relação na qual vai se transformar a entidade fraca vai receber uma chave da sua entidade forte, que identifique a qual entidade forte está ligada. Esse é o conceito de chave que faltava para nós, o de chave estrangeira. **Chave estrangeira é um conjunto de um ou mais atributos de uma relação que fazem referência à chave primária de outra relação, ou até mesmo à própria (nos auto-relacionamentos).** Nosso DER não tem entidades fracas, então vamos ao exemplo do relacionamento entre Funcionário e Dependente.



Fazendo esse mapeamento, teremos:

Funcionario (matrFunc, nomeFunc, salário, cargo)

Dependente (idDep, nomeDep, dataNac, sexo, matrFunc)

Viram que a Relação Dependente ganhou um atributo no final, chamado de matrFunc? Esse atributo é a **chave estrangeira**, pois vai identificar de qual funcionário é esse dependente. Além do mais, esse atributo deve ser chave primária na tabela de origem (Funcionario) para ser chave estrangeira na tabela que recebeu a chave (Dependente). Claro, se a chave primária de Funcionario fosse composta de dois atributos, a chave estrangeira em Dependente seriam esses dois atributos.

Passo 3 – Mapeamento dos relacionamentos 1:N: Nesse tipo de relacionamento, também devemos exportar chave estrangeira. Vamos a um exemplo:



Os atributos não estão listados no DER, mas vamos imaginar que fazendo as relações, de acordo com o passo 1, teremos o seguinte:

Funcionário (Matr, Nome, DtNascimento, sexo)

Departamento (CodDepartamento, NomeDepartamento, Localização)

Bem, nesse passo, devemos deslocar a chave primária da entidade que está próximo ao 1 para a entidade que está próximo ao N. Trocando em miúdos, transferimos a chave primária de Departamento, para ser chave estrangeira de Funcionario. As relações ficam assim:

Funcionário (Matr, Nome, DtNascimento, sexo, CodDepartamento)

Departamento (CodDepartamento, NomeDepartamento, Localização)

Passo 4 – Mapeamento dos relacionamentos N:M: Esse tem no nosso modelo dos DVDs. Aqui o que fazemos, em suma é o seguinte. Não dá para passar as chaves primárias de uma entidade para outra, então criamos uma nova Relação, que vai fazer o meio de campo entre as duas entidades. No nosso exemplo, temos que um amigo empresta N DVDs, e um DVD pode ser emprestado por N amigos. Então, vou criar uma relação, que vou chamar de Empréstimo, que irá viabilizar o relacionamento entre amigos e DVDs. Como crio essa relação? Faço o seguinte: pego a chave primária de cada entidade envolvida, e jogo na nova relação, dessa forma:

Empréstimo (CPF, IdDVD)

Na sequência verifico se o relacionamento possuía atributos no DER. Se possuía, esses atributos também vão para a nova relação. Fica assim:

Empréstimo (CPF, IdDVD, DataEmprest, DataPrev, DataDevol)

Por fim, defino a chave primária da minha relação Empréstimo. Via de regra, os atributos que vieram como chaves estrangeiras formam a chave primária composta da nova relação. No nosso caso, isso não vai ser possível, pois eu suponho que um amigo (chato) pode emprestar o mesmo DVD mais de uma vez. Então, tenho duas opções: apelo para o velho ID, ou acrescento o campo DataEmprest na chave (afinal, não pode o mesmo amigo emprestar o mesmo DVD na mesma data mais de uma

vez). Vou usar o Id, para simplificar. Vamos ver como ficam nossas três relações.

DVD (IdDVD, TituloPort, TituloIng, Tipo, Gênero, Ano)

Amigo (CPF, Nome, Endereço)

Empréstimo (IdEmprest, CPF, IdDVD, DataEmprest, DataPrev, DataDevol)

Tem diversas outras formas de definir essas chaves, mas isso não interessa para nós, vamos nos ater às regras mais usadas.

Passo 5 – Mapeamento dos Relacionamentos 1:1: de forma bem simples, escolhemos qualquer um dos relacionamentos e passamos sua chave primária para a outra relação como chave estrangeira. Por exemplo, naquele relacionamento marido e mulher, ou pegamos a chave primária de mulher e passamos para marido, ou vice versa. Como o relacionamento é 1:1, não faz diferença. Pelo menos no Modelo Relacional é bom ter a ilusão que é a mulher que está vinculada ao marido!!

Passo 6 – Mapeamento dos atributos multivalorados: temos no nosso exemplo isso também. Existem diversas soluções, e eu vou dar uma. Cada atributo pode virar uma Relação, com um relacionamento N:M para a relação de onde saiu. Por exemplo, na entidade DVD/Bly-ray tinha os atributos atores e diretores, ambos multivalorados. Como fazemos o mapeamento?

Primeiro vamos transformá-los em relações. Eles poderiam ficar assim:

Ator (IdAtor, NomeAtor)

Diretor (IdDiretor, NomeDiretor)

Bem, vejam que eu já pulei passos, e já criei campos Id para serem as chaves primárias dessas relações. Qual a cardinalidade do relacionamento entre Ator e DVD? E Diretor e DVD? Bem, um DVD tem a

participação de N atores, e um ator pode participar de N DVDs (ou seja, de N filmes, séries etc). E Diretor? A mesma coisa, concordam? Bem, se tem um relacionamento N:M, aplicamos o passo 4. Vou criar mais duas relações, uma chamada AtorParticipa e outra chamada DiretorDirige. No final, aplicando a exportação de chaves, deve ficar assim:

AtorParticipa (IdDVD, IdAtor)

DiretorDirige (IdDVD, IdDiretor)

Por fim, podemos fazer a mesma coisa com o atributo multivalorado telefone. Mas uma outra alternativa é o que vou adotar. Bem, meus amigos têm no máximo três telefones. E se tiverem mais? Do quarto em diante não interessa para mim. Então vou colocar três campos em Amigo para guardar até três telefones. Finalizando nosso modelo, ele fica assim:

DVD (IdDVD, TituloPort, TituloIng, Tipo, Gênero, Ano)

Amigo (CPF, Nome, Endereço, Tel1, Tel2, Tel3)

Empréstimo (IdEmprest, CPF, IdDVD, DataEmprest, DataPrev, DataDevol)

Ator (IdAtor, NomeAtor)

Diretor (IdDiretor, NomeDiretor)

AtorParticipa (IdDVD, IdAtor)

DiretorDirige (IdDVD, IdDiretor)

Pronto! Acabamos nosso mapeamento do MER para o Modelo Relacional. Transformamos um DER que tinha 2 entidades em um modelo com 7 relações, que pode agora ser implementado em um Banco de Dados Relacional. Guardem esse modelo, que depois passaremos ele para um modelo físico do Banco de Dados.

2.4. RESTRIÇÕES DE INTEGRIDADE

O Modelo Relacional traz consigo uma série de restrições, que são características importante do Modelo. Entre elas, as mais populares são as restrições de integridade. As restrições de integridade são regras aplicadas pelo Banco de Dados para garantir que o Banco permaneça íntegro, exato e consistente. Em outras palavras, que o Banco de Dados reflita a realidade modelada. Algumas das restrições de integridade mais importantes são:

- Integridade de domínio (ou restrição de domínio): visa garantir que os valores que cada atributo receba esteja dentro do seu domínio. Por exemplo, um campo sexo deve receber M ou F, um campo data deve receber uma data válida, e assim sucessivamente.
- Integridade de chave (ou restrição de chave): Impede que uma chave primária se repita.
- Integridade de Entidade (ou restrição de entidade): Impede que uma chave primária receba em qualquer de seus campos o valor NULL.
- Integridade de vazio: subtipo da integridade de domínio, verifica se um campo pode ou não receber NULL.
- Integridade referencial: agora que estudamos chaves estrangeiras, podemos entender melhor esse conceito. Essa integridade visa garantir que o valor de um campo que é chave estrangeira em uma tabela exista na chave primária na tabela de origem. Por exemplo, imaginem que na tabela Empréstimo do nosso modelo, alguém tente incluir um empréstimo com um número de CPF que não existe na tabela Amigo. Se isso acontecer, o Banco ficará inconsistente, pois quando procurarmos de quem é aquele CPF na tabela de amigo, verificaremos que ele não existe. O SGBD ou a aplicação devem garantir que isso não aconteça. Outra

possível violação à integridade referencial é quando tentamos excluir um registro que está em outras tabelas como chave estrangeira. Por exemplo, imaginem que eu tente excluir determinado DVD, sendo que aquele IdDVD está na tabela Empréstimo. Ora, se eu fizer isso, não vai ser possível saber a qual DVD aquele código diz respeito. Assim, deve ser garantida a integridade de referencial também nesse caso. Na pior das hipóteses, na chave estrangeira pode haver o valor NULL.

7. (ESAF/AFRF/SRF 2002) Analise as seguintes afirmações relativas às regras de integridade do modelo, no projeto de banco de dados:

I. Nenhum campo que participa da chave primária de uma tabela básica pode aceitar valores nulos.

II. Pode existir na chave estrangeira um valor que não exista na tabela na qual ela é chave primária.

III. Se uma determinada tabela T1 possui uma chave estrangeira, a qual é chave primária em uma tabela T2, então ela deve ser igual a um valor de chave primária existente em T2 ou ser nula.

IV. Uma tabela só é acessível por um campo se este for chave primária.

Indique a opção que contenha todas as afirmações verdadeiras.

a) I e II

b) II e III

c) III e IV

d) I e III

e) II e IV

Comentários:

I. Sim, já vimos essa restrição de integridade. Correto.

II. Não, vimos que a chave estrangeira

III. Sim, conforme acabamos de definir

IV. Não, a chave primária é uma boa forma de acessar os registros, mas não é o único.

Gabarito: Letra d

8. (ESAF/AFRF/SRF 2003) Considerando os graus de relacionamentos no desenvolvimento de banco de dados, é correto afirmar que

a) uma tabela com uma chave primária, que receba com frequência valores nulos, poderá receber um relacionamento muitos-para-muitos com outra tabela, desde que esta seja chave estrangeira na segunda tabela.

b) pode-se definir um relacionamento muitos-para-muitos entre duas tabelas criando-se uma terceira tabela e definindo-se os relacionamentos um-para-muitos e muitos-para-um entre esta última e as duas primeiras tabelas.

c) uma forma de se obter um relacionamento um-para-um entre duas tabelas é criando-se uma terceira tabela e definindo-se os relacionamentos um-para-muitos e muitos-para-um entre esta última e as duas primeiras tabelas.

d) uma tabela com uma chave primária, que receba com frequência valores repetidos, poderá receber um relacionamento muitospara- muitos com uma segunda tabela, criando- se uma

terceira tabela e definindo-se os relacionamentos um-para-muitos e muitospara- um entre esta última e as duas primeiras tabelas.

e) parte das informações armazenadas em um banco de dados relacional, que possua relacionamentos um-para-um entre pelo menos uma de suas tabelas, perde a garantia de ser logicamente acessível.

Comentários:

a) Primeiro a redação ficou confusa. Quem recebe muitos valores nulos, a tabela ou a chave primária? Se for a chave, conversa encerrada, sabemos que não pode. Continuando a questão, se uma tabela de cardinalidade N:M se relaciona com outra, já sabemos que precisamos de uma outra tabela aí para fazer o meio de campo. Item incorreto.

b) Exato, foi como fizemos entre Amigo e DVD, criamos a Relação Empréstimo, que tem cardinalidade 1:N com Amigo (um amigo efetua muitos empréstimo, e um empréstimo é feito por um amigo) e também 1:N com DVD (um empréstimo se refere a um DVD, mas um DVD pode ser emprestado N vezes). Item correto.

c) Entre 1:1 não precisa criar uma terceira tabela.

d) Nem a chave muito menos a tabela pode receber valores repetidos. Item errado, absurdo, abmudo, abcego!!!

e) Nada a ver com nada também, isso é kaiser quente misturado com caninha Tatuzinho!!

Gabarito: Letra b

9. (ESAF/Analista de TI/Sefaz-CE 2007) Quando o atributo chave primária de uma entidade é exportado para outra entidade geram-se

a) erros.

b) chaves estrangeiras.

c) views.

d) chaves primárias duplicadas.

e) agregações.

Comentários:

Essa ficou fácil agora, não é? Chave estrangeira!

Gabarito: Letra b

10. (ESAF/Analista de TI/Sefaz-CE 2007) Em Abordagem Relacional

a) uma chave relacionada é uma coluna cujos valores distinguem atributos de relacionamentos.

b) a chave estrangeira é o mecanismo que permite a implementação de relacionamentos em um banco de dados relacional.

c) a chave estrangeira é uma coluna ou uma combinação de colunas cujos valores não aparecem na chave primária de uma tabela.

d) uma chave primária é uma linha ou uma combinação de linhas cujos valores distinguem uma coluna das demais dentro de uma tabela.

e) a chave estrangeira é uma linha ou uma combinação de linhas cujos valores necessariamente aparecem na chave primária de uma tabela.

Comentários:

a) Credo!!!!!! What the fuc&%%\$# hell is that??? Não tenho idéia do que o examinador tentou falar.

b) Agora sim, uma informação legal. Lembra que eu disse para vocês que os relacionamentos são o pulo do gato desse modelo? Pois é,

pelos mapeamentos vocês devem ter notado que a forma de implementar esses relacionamentos é pelas chaves estrangeiras. Por exemplo, o relacionamento entre empregado e dependente se deu passando a matrícula do empregado como chave estrangeira para dependente. E assim fizemos com diversas relações. Aleluia Glória a Deus Meu Pai, uma definição inteligente da banca.

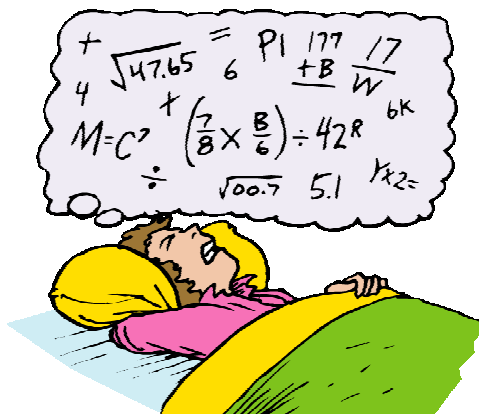
c) Aparecem sim na chave primária, ou podem ser nulas.

d) Chave nenhuma é linha ou combinação de linhas. É coluna ou combinação de colunas.

e) ver letra (d)

Gabarito: Letra b

2.5. ÁLGEBRA RELACIONAL



Muita calma nessa hora!!! Não se assustem com o título. Apesar do nome Álgebra assustar um pouco, esse assunto não é dos mais pesados, mas é muito importante para entendermos SQL.

A Álgebra Relacional é uma linguagem de consulta formal, porém procedimental, ou seja, o usuário dá as instruções ao sistema para que o mesmo realize uma seqüência de operações na base de dados para

calcular o resultado desejado. A álgebra relacional é uma forma de cálculo sobre conjuntos ou relações.

Mas professor, não era para não ficar assustado? Ok, vamos simplificar. O que quis dizer com esses conceitos é que, como o modelo relacional é um modelo matemático, a álgebra relacional é a base para que possamos recuperar informações nas nossas relações, ou trocando em miúdos, para que possamos acessar as informações no nosso Banco de Dados.

Existem diversas operações em álgebra relacional, entre as quais: seleção, projeção, renomear, produto cartesiano, união e diferença entre conjuntos, interseção de conjuntos, junção natural. Veremos as principais delas, e implicitamente já estaremos aprendendo um pouco de SQL, mas quando vermos os comandos do SQL tudo fica mais fácil.

Vamos ver as duas primeiras, mais usadas, mais comentadas, mais fáceis. A seleção e a projeção.

A seleção é utilizada para selecionar um conjunto de tuplas (linhas) de uma relação. Assim, ela funciona como um filtro para selecionar tuplas que atendem a determinado critério. Vamos voltar a uma relação para tomar como exemplo.

CPF	Nome	Nascimento	Endereco	Cidade	Estado
111222	Alberto	01/01/1975	Rua 17 n 76	Manaus	AM
123445	Ana	03/05/1947	Rua A quadra 1	Brasília	DF
345321	Luiza	12/10/1990	Quadra L casa 6	Salvador	BA
237980	Francisco	17/02/1980	Cond Maresia ap 301	Fortaleza	CE
567987	Guilherme	25/08/1970	Rua 101, n 303	Maceió	AL
563029	Patricia	15/12/1989	Av das Torres n 130	São Paulo	SP
908123	Milena	28/11/1974	Quadra H ap 304	Boa Vista	RR
985209	Andrea	20/05/1946	Rua das Lamentações, 120	Rio Branco	AC

Temos aí uma relação Cliente que usamos faz pouco tempo. Uma operação de seleção visa retornar determinadas tuplas que satisfaçam um critério. Vamos supor, com base na relação acima, que eu deseje

consultar todas as tuplas em que a data de nascimento seja superior a 1/1/1980. A operação em álgebra relacional que representa isso é a seguinte:

σ Nascimento > 1/1/1980 (Cliente)

Simplesmente essa operação em álgebra relacional significa: retorne as tuplas da Relação Cliente em que o atributo nascimento seja maior que 1980.

Essa operação naquela relação retornaria outra relação, só com as tuplas que obedecem ao critério:

CPF	Nome	Nascimento	Endereco	Cidade	Estado
345321	Luiza	12/10/1990	Quadra L casa 6	SaLvador	BA
237980	Francisco	17/02/1980	Cond Maresia ap 301	Fortaleza	CE
563029	Patricia	15/12/1989	Av das Torres n 130	São Paulo	SP

Assim, representamos a seleção dessa forma

σ <condição> (Relação), onde:

σ (sigma) é o símbolo da operação de seleção;

<condição> é o critério de seleção;

(Relação) é a relação na qual a operação vai agir.

É importante frisar que essa operação não afeta a relação original Cliente, que fica do jeitinho que estava. Simplesmente uma nova relação é retornada como resultado da operação.

Por selecionar linhas em uma Relação, essa operação é também denominada particionamento horizontal, uma vez que a operação dá um corte horizontal na relação, retornando algumas tuplas e deixando as demais.

A segunda operação importante é a projeção. **A operação de projeção seleciona certas colunas da tabela (relação) e descarta**

outras. Ela é a mais simples de todas. Vamos ver como ela se aplica na nossa relação Cliente.

π CPF, Nome, Estado (Cliente)

Essa operação retorna todas as tuplas da relação cliente, mas somente com os atributos selecionado, retornaria algo do tipo:

CPF	Nome	Estado
111222	Alberto	AM
123445	Ana	DF
345321	Luiza	BA
237980	Francisco	CE
567987	Guilherme	AL
563029	Patricia	SP
908123	Milena	RR
985209	Andrea	AC

Agora temos todas as tuplas, mas somente com os atributos selecionados. Sua forma geral é:

π <atributos> (Relação), onde:

π (PI) é o símbolo da operação de projeção;

<atributos> é a lista de atributos escolhidos;

(Relação) é a relação na qual a operação vai agir.

Esses dois operadores são considerados operadores **unários**, pois atuam em apenas uma relação. E se eu quiser em uma operação só juntar as duas, eu posso? Claro que sim, e essa parceria é a base de todas as consultas que vocês vão aprender a fazer em Bancos de Dados. Vamos ver a junção das duas operações, supondo agora que eu quero CPF, Nome e Estado dos Clientes nascidos após 1/1/1980.

π CPF, Nome, Estado (σ Nascimento > 1/1/1980 (Cliente))

Pronto, temos aí nossa consulta. Primeiro é resolvido o que está dentro dos parênteses, ou seja, é retornada uma relação com as tuplas que obedecem à condição estabelecida, ou seja, que o atributo

Nascimento é maior que 1/1/1980. Nessa relação retornada é feita uma operação de projeção, para retornar apenas os campos listados. O resultado final dessas operações é:

CPF	Nome	Estado
345321	Luiza	BA
237980	Francisco	CE
563029	Patricia	SP

Por selecionar colunas em uma Relação, essa operação é também denominada **particionamento vertical**, uma vez que a operação dá um corte vertical na relação, retornando algumas colunas e deixando os demais.

Essas operações vão nos ajudar demais no SQL. E acreditem, é mais fácil no SQL do que na álgebra relacional.

Posso apresentar mais um hoje?

Vamos à operação produto cartesiano. Ela é bem simples de entender. É uma operação binária (atua em duas relações) que une cada elemento de uma relação a um elemento da outra relação. Vamos a um exemplo, considerando duas Relações, R1 e R2, cada uma com somente um atributo (o nome desse atributo foi oculto no exemplo, porque é irrelevante)

<table><tr><th>R1</th></tr><tr><td>A</td></tr><tr><td>B</td></tr></table>	R1	A	B	X	<table><tr><th>R2</th></tr><tr><td>C</td></tr><tr><td>D</td></tr><tr><td>E</td></tr></table>	R2	C	D	E	<table><tr><th>R1 X R2</th></tr><tr><td>AC</td></tr><tr><td>AD</td></tr><tr><td>AE</td></tr><tr><td>BC</td></tr><tr><td>BD</td></tr><tr><td>BE</td></tr></table>	R1 X R2	AC	AD	AE	BC	BD	BE
R1																	
A																	
B																	
R2																	
C																	
D																	
E																	
R1 X R2																	
AC																	
AD																	
AE																	
BC																	
BD																	
BE																	

Então a operação de Produto Cartesiano, representado por X, juntou cada elemento de R1 com cada elemento de R2, tendo como resultado a Relação R1 X R2.

Depois veremos outras operações importantes, mas é melhor seguir junto com a aprendizagem do SQL.

2.6. LINGUAGENS DE BANCO DE DADOS

Bem, como não gosto de deixar pontos soltos, vamos pensar um pouco o que faremos com nosso Modelo Relacional, que fechamos lá atrás. Como já conversamos, o próximo passo é elaborar o projeto físico do nosso Banco de Dados. Mas o que é o projeto físico, afinal?

No frigar dos ovos, nosso projeto físico é simplesmente pegar nosso modelo relacional pronto e transformar em um Banco de Dados, dentro de um SGBD. Mas aí começa uma questão: como vamos informar ao SGBD que nosso Banco de Dados tem tais tabelas, com tais campos, tais chaves primárias etc etc etc.

É aí que entram as linguagens que foram criadas para “dialogar” com o Banco de Dados. Vamos ver as principais:

- DDL (Data Definition Language): Quando você cria um banco de dados, ele inicialmente está “vazio”; antes de começar a consultar e alterar dados é preciso definir onde e como as informações serão gravadas dentro do novo banco; então você cria diversas tabelas explicitando o tipo de dados de cada campo, as chaves estrangeiras, os índices, as regras e etc. Estes comandos de criação e alteração de estrutura são os comandos de DEFINIÇÃO dos dados, pois definem como os dados serão armazenados; em inglês são chamados de: Data Definition Language (DDL).

- DML (Data Manipulation Language): Depois que você criou suas tabelas, definiu relacionamentos, índices e etc., é hora de manipular seus dados. O que quer dizer isso? Quer dizer que você irá fazer operações típicas nas tabelas, como inserção de informações, alterações, exclusões e consultas. Para isso temos a linguagem de manipulação dos dados, chamada de DML.
- DCL (Data Control Language): Com o banco de dados pronto e rodando é importante definir quem poderá acessá-lo, enfim, precisamos definir a segurança do seu banco. Em inglês, os comandos responsáveis pelo controle dos dados são chamados de Data Control Language (DCL).

Agora, uma surpresa para vocês. Sabe quem implementa todas essas linguagens nos Banco de Dados Relacionais? O nosso querido SQL. Structured Query Language, ou Linguagem de Consulta Estruturada ou SQL, é uma linguagem de pesquisa declarativa para banco de dados relacional (base de dados relacional). Muitas das características originais do SQL foram inspiradas na álgebra relacional.

Na próxima aula começaremos a ver de verdade o SQL. Precisamos aprender uma quantidade razoável de comandos e praticar um pouco. Mas a base dele nós já vimos, que são as operações de álgebra relacional.

3. QUESTÕES COMENTADAS

11. (ESAF/ATI/SUSEP 2010) Um modelo de banco de dados relacional deve conter no mínimo a definição de:

- a) tabelas, colunas das tabelas e restrições de integridade.**
- b) títulos, colunas dos atributos e restrições de integridade.**
- c) títulos, colunas das tabelas e restrições de manutenção.**

d) tabelas, relações entre linhas das tabelas e opções de integridade.

e) associações de restrição, colunas referenciadas e restrições de desempenho.

Comentários:

Temos na letra (a) os elementos importantes de um Banco de Dados Relacional, ou seja, de um Banco de Dados que implementa o modelo Relacional, que são as tabelas (Relações), as colunas (atributos) e as restrições de integridade. As outras alternativas têm um monte de firulas, tipo títulos, restrições de manutenção e por ai vai.

Gabarito: Letra a

12. (ESAF/ATI/SUSEP 2010) Em Sistemas Gerenciadores de Bancos de Dados (SGBD), existem as seguintes categorias de restrições de integridade:

a) Integridade de domínio, Integridade de vazio, Integridade de chave, Integridade referencial.

b) Integridade de acesso, Integridade de entrada, Integridade de saída, Integridade referencial.

c) Integridade de domínio, Integridade de completude, Integridade de chave, Integridade posicional.

d) Integridade de cardinalidade, Integridade de vazio, Integridade de autorização de acesso, Integridade associativa.

e) Integridade de generalização/especialização, Integridade de usuários, Integridade de chave, Integridade referencial.

Comentários:

Questão também vista e debatida na aula. Temos na letra (a) a resposta para nossa questão.

Gabarito: Letra a

13. (ESAF/ATI/SUSEP 2006) Em um Banco de Dados Relacional:

a) uma relação está na 1FN (primeira forma normal) se nenhum domínio contiver valores atômicos.

b) uma Chave Primária corresponde ao Identificador único de uma determinada relação. Em uma relação pode haver mais que uma coluna candidata a chave primária.

c) as colunas que irão compor as Chaves Primárias devem ser inicializadas com valores nulos.

d) em uma tabela existirão tantas Chaves Primárias quantas forem as colunas nela existentes.

e) uma Chave Externa é formada por uma coluna de uma tabela que se referencia a uma Coluna qualquer de outra tabela. Essas colunas, na tabela destino, não aceitam valores nulos. Uma tabela destino pode ter apenas uma Chave Externa.

Comentários:

a) Não falamos sobre formais normais, só veremos na próxima aula, junto com SQL. Mas o item está errado.

b) Exato, define bem chave primária e ainda conceitua chave candidata, conforme vimos.

c) Pelo contrário, não pode aceitar nulo. Vejam que a ESAF gosta de cobrar isso.

d) Não, cada tabela só tem uma chave primária.

e) Entendam por chave externa a chave estrangeira. A chave estrangeira não se referencia a uma coluna qualquer de outra tabela, e sim à chave primária dessa tabela. As chaves estrangeiras aceitam valores nulos. Por fim, e muito importante, uma tabela pode ter mais de

uma chave estrangeira. É o caso da nossa tabela Empréstimo, que tem chave estrangeira vinda de Amigo (CPF) e outra vinda de DVD (IdDVD).

Gabarito: Letra b

14. (ESAF/AFRF/SRF 2005) Com relação aos conceitos básicos de banco de dados, é correto afirmar que:

a) a chave primária é um atributo de uma tabela que, mesmo com valores nulos, identifica univocamente uma coluna.

b) o modelo relacional refere-se à visualização física e não lógica dos dados. Está relacionado ao nível conceitual interno. A teoria relacional não diz nada sobre o nível externo, preocupa-se somente com o armazenamento e manipulação dos dados executados pelo SGBD.

c) chaves estrangeiras são os elos de ligação entre as tabelas. Uma coluna definida como chave estrangeira deve ser chave primária em outra tabela.

d) um banco de dados relacional é um conjunto de arquivos seqüenciais que são acessados e modificados por operações que manipulam a álgebra relacional. Tais operações só podem ser executadas se atenderem à regra da primeira forma normal, devendo-se manipular apenas um dado de cada vez.

e) uma coluna definida como chave-estrangeira em uma tabela-destino não pode aceitar valores nulos e, essa mesma tabela-destino pode ter uma e somente uma coluna definida como chave-estrangeira.

Comentários:

- a) Nada de valores nulos na chave primária
- b) O modelo relacional é lógico, e não físico, conforme estudamos.
- c) Exatamente

d) Um Banco de Dados Relacional não é um conjunto de arquivos seqüenciais, é um conjunto de tabelas ou relações, com os seus relacionamentos.

e) Já vimos que está duplamente errada a afirmativa.

Gabarito: Letra c

Amigos, fico por aqui nessa aula, estamos avançando a passos largos, e estou otimista que veremos com muita calma tudo o que é mais relevante para vocês fazerem uma excelente prova. Deixo uma pequena tirinha para distrair um pouco.

Grande abraço

Leonardo Lima

leonardolima@estrategiaconcursos.com.br

**Querido suporte,
Obrigado por criar minha conta tão rápido.
Mas apesar de engraçado, como vou vender
nossos produtos com um e-mail como
vadia@accorhotel.com ?**

**Desde já agradeço.
Valter Dias**

**Querido Sr Dias
Infelizmente as contas são geradas a partir
das iniciais de seu nome. Não podemos mudar.
Pode acreditar em mim.**

**Saudações,
Vitor Adonis viado@accorhotel.com**

4. RESUMO

- O Modelo Relacional representa os dados em um Banco de Dados como um conjunto de tabelas, que no modelo são denominadas Relações. Cada Relação terá um nome, que será único, e um conjunto de atributos, com seus respectivos nomes e domínios.
- O grau da Relação é simplesmente o número de atributos que a mesma possui
- O instante (snapshot) da Relação é a fotografia de um determinado momento da Relação.
- Características da Relação:
 - Ordenação: a ordem das tuplas e dos atributos não importam para a Relação.
 - Atomicidade: todos os atributos devem ter valores atômicos, ou seja, únicos.
 - Identidade: cada atributo de uma Relação tem um nome que é único para aquela Relação.
 - Unicidade: todas as tuplas devem ser únicas
- Chave: conjunto de um ou mais atributos que determinam a unicidade de cada tupla (registro). Vamos ver que chave é um gênero que comporta muitas espécies diferentes.
- Superchave é um conjunto de um ou mais atributos que, tomados coletivamente, nos permitem identificar de maneira unívoca uma entidade em um conjunto de entidades
- Chaves candidatas. São superchaves de tamanho mínimo, candidatas a serem chaves primárias da Relação Ou seja, são um

atributo ou conjunto de atributos que permitem identificar de forma inequívoca qualquer tupla dessa relação, mas que não pode ser reduzido sem perder qualidade.

- Toda superchave que tem apenas um atributo é uma chave candidata
- Entre as chaves candidatas possíveis, devemos escolher para cada relação uma, e somente uma, chave primária. Chave primária nada mais é um conjunto de um ou mais campos, cujos valores, considerando a combinação de valores de todos os campos da tupla, nunca se repetem e que podem ser usadas como um índice para os demais atributos da relação, escolhida entre as chaves candidatas.
- Em chaves primárias, não pode haver valores nulos nem repetição de tuplas.
- Chave estrangeira é um conjunto de um ou mais atributos de uma relação que fazem referência à chave primária de outra relação, ou até mesmo à própria (nos auto-relacionamentos)
- As restrições de integridade são regras aplicadas pelo Banco de Dados para garantir que o Banco permaneça íntegro, exato e consistente.
- Integridade de domínio (ou restrição de domínio): visa garantir que os valores que cada atributo receba esteja dentro do seu domínio. Por exemplo, um campo sexo deve receber M ou F, um campo data deve receber uma data válida, e assim sucessivamente.
- Integridade de chave (ou restrição de chave): Impede que uma chave primária se repita.
- Integridade de Entidade (ou restrição de entidade): Impede que uma chave primária receba em qualquer de seus campos o valor NULL.

- Integridade de vazio: subtipo da integridade de domínio, verifica se um campo pode ou não receber NULL.
- Integridade referencial: garante que chaves estrangeiras estejam relacionadas a chaves existentes nas tabelas de origem.
- A álgebra relacional é uma forma de cálculo sobre conjuntos ou relações.
- A seleção (particionamento horizontal) é utilizada para selecionar um conjunto de tuplas (linhas) de uma relação. Assim, ela funciona como um filtro para selecionar tuplas que atendem a determinado critério.
- A operação de projeção (particionamento vertical) seleciona certas colunas da tabela (relação) e descarta outras
- Produto cartesiano é uma operação binária (atua em duas relações) que une cada elemento de uma relação a um elemento da outra relação.
- DDL (Data Definition Language): Quando você cria um banco de dados, ele inicialmente está "vazio"; antes de começar a consultar e alterar dados é preciso definir onde e como as informações serão gravadas dentro do novo banco; então você cria diversas tabelas explicitando o tipo de dados de cada campo, as chaves estrangeiras, os índices, as regras e etc. Estes comandos de criação e alteração de estrutura são os comandos de DEFINIÇÃO dos dados, pois definem como os dados serão armazenados; em inglês são chamados de: Data Definition Language (DDL).
- DML (Data Manipulation Language): Depois que você criou suas tabelas, definiu relacionamentos, índices e etc., é hora de manipular seus dados. Quer dizer que você irá fazer operações típicas nas tabelas, como inserção de informações, alterações, exclusões e

consultas. Para isso temos a linguagem de manipulação dos dados, chamada de DML.

- DCL (Data Control Language): Com o banco de dados pronto e rodando é importante definir quem poderá acessá-lo, enfim, precisamos definir a segurança do seu banco. Em inglês, os comandos responsáveis pelo controle dos dados são chamados de Data Control Language (DCL).

5. LISTA DAS QUESTÕES APRESENTADAS

1. (ESAF/Analista Desenvolvimento de Sistemas/ANA 2009)
O modelo de dados baseado numa coleção de tabelas que representam dados e as relações entre eles é denominado modelo

- a) relacional.**
- b) entidade/relacionamento.**
- c) baseado em objetos.**
- d) de dados semiestruturados.**
- e) objeto/relacionamento.**

2. (ESAF/AFRF/SRF 2002) Em um banco de dados relacional, os objetos que realmente armazenam os dados são

- a) as chaves primárias.**
- b) os relacionamentos.**
- c) as tabelas.**
- d) as transações.**
- e) os procedimentos armazenados.**

3. (ESAF/Analista Desenvolvimento Sistemas/ANA 2009) Um conjunto de um ou mais atributos, tomados coletivamente, para identificar unicamente uma tupla numa relação, é denominado

- a) chave assimétrica.**
- b) chave simétrica.**
- c) superchave.**
- d) chave secundária.**
- e) chave de tupla.**

4. (ESAF/TRF/SRF 2006) Analise as seguintes afirmações relacionadas a Bancos de Dados:

I. Em uma tabela, quando existir uma combinação de colunas que sirva para identificar todos os registros dessa tabela, essa combinação poderá ser escolhida como uma chave primária composta.

II. Em um banco de dados, quando se deseja garantir que, em uma coluna ou combinações de coluna, a qualquer momento, nenhum par de linhas da tabela deva conter o mesmo valor naquela coluna ou combinação de colunas, é necessário definir uma chave primária.

III. Uma das regras da integridade do modelo relacional é possibilitar que um atributo que participe da chave primária de uma relação básica aceite um e somente um valor nulo.

IV. Normalização é o processo de se reunir todos os dados que serão armazenados em um certo banco de dados e concentrá-los em uma única tabela.

Indique a opção que contenha todas as afirmações verdadeiras.

- a) II e III**

- b) I e II**
- c) III e IV**
- d) I e III**
- e) II e IV**

5. (ESAF/ACE-Sistemas/TCU 2002) Analise as seguintes afirmações relativas a Banco de Dados:

I. Uma chave primária não pode desempenhar a função de identificação única;

II. Um modelo conceitual de banco de dados representa a estrutura de dados de um Banco de Dados com os recursos e particularidades de um Sistema de Gerenciamento de Banco de Dados específico;

III. Entidade pode ser definida como um objeto que existe no mundo real, com uma identificação distinta e com significado próprio;

IV. Uma das regras da integridade do modelo relacional afirma que nenhum campo que participe da chave primária de uma tabela básica pode aceitar valores nulos.

Indique a opção que contenha todas as afirmações verdadeiras.

- a) I e II**
- b) II e III**
- c) III e IV**
- d) I e III**
- e) II e IV**

6. (ESAF/TTN-Informática/SRF 1998) Em relação aos bancos de dados é correto afirmar que

- a) a chave primária define uma ordem padrão para a ordenação dos campos de um registro**
- b) os campos lógicos podem armazenar strings de caracteres quaisquer**
- c) as colunas das tabelas que compõem um banco de dados são chamadas campos e as linhas são chamadas registros**
- d) a chave primária só pode ser formada por um único campo**
- e) os bancos de dados relacionais são também chamados de bancos de dados simples**

7. (ESAF/AFRF/SRF 2002) Analise as seguintes afirmações relativas às regras de integridade do modelo, no projeto de banco de dados:

I. Nenhum campo que participa da chave primária de uma tabela básica pode aceitar valores nulos.

II. Pode existir na chave estrangeira um valor que não exista na tabela na qual ela é chave primária.

III. Se uma determinada tabela T1 possui uma chave estrangeira, a qual é chave primária em uma tabela T2, então ela deve ser igual a um valor de chave primária existente em T2 ou ser nula.

IV. Uma tabela só é acessível por um campo se este for chave primária.

Indique a opção que contenha todas as afirmações verdadeiras.

- a) I e II**
- b) II e III**

c) III e IV

d) I e III

e) II e IV

8. (ESAF/AFRF/SRF 2003) Considerando os graus de relacionamentos no desenvolvimento de banco de dados, é correto afirmar que

a) uma tabela com uma chave primária, que receba com frequência valores nulos, poderá receber um relacionamento muitos-para-muitos com outra tabela, desde que esta seja chave estrangeira na segunda tabela.

b) pode-se definir um relacionamento muitos-para-muitos entre duas tabelas criando-se uma terceira tabela e definindo-se os relacionamentos um-para-muitos e muitos-para-um entre esta última e as duas primeiras tabelas.

c) uma forma de se obter um relacionamento um-para-um entre duas tabelas é criando-se uma terceira tabela e definindo-se os relacionamentos um-para-muitos e muitos-para-um entre esta última e as duas primeiras tabelas.

d) uma tabela com uma chave primária, que receba com frequência valores repetidos, poderá receber um relacionamento muitospara- muitos com uma segunda tabela, criando- se uma terceira tabela e definindo-se os relacionamentos um-para-muitos e muitospara- um entre esta última e as duas primeiras tabelas.

e) parte das informações armazenadas em um banco de dados relacional, que possua relacionamentos um-para-um entre pelo menos uma de suas tabelas, perde a garantia de ser logicamente acessível.

9. (ESAF/Analista de TI/Sefaz-CE 2007) Quando o atributo chave primária de uma entidade é exportado para outra entidade geram-se

- a) erros.**
- b) chaves estrangeiras.**
- c) views.**
- d) chaves primárias duplicadas.**
- e) agregações.**

10. (ESAF/Analista de TI/Sefaz-CE 2007) Em Abordagem Relacional

a) uma chave relacionada é uma coluna cujos valores distinguem atributos de relacionamentos.

b) a chave estrangeira é o mecanismo que permite a implementação de relacionamentos em um banco de dados relacional.

c) a chave estrangeira é uma coluna ou uma combinação de colunas cujos valores não aparecem na chave primária de uma tabela.

d) uma chave primária é uma linha ou uma combinação de linhas cujos valores distinguem uma coluna das demais dentro de uma tabela.

e) a chave estrangeira é uma linha ou uma combinação de linhas cujos valores necessariamente aparecem na chave primária de uma tabela.

11. (ESAF/ATI/SUSEP 2010) Um modelo de banco de dados relacional deve conter no mínimo a definição de:

- a) tabelas, colunas das tabelas e restrições de integridade.**

- b) títulos, colunas dos atributos e restrições de integridade.**
- c) títulos, colunas das tabelas e restrições de manutenção.**
- d) tabelas, relações entre linhas das tabelas e opções de integridade.**
- e) associações de restrição, colunas referenciadas e restrições de desempenho.**

12. (ESAF/ATI/SUSEP 2010) Em Sistemas Gerenciadores de Bancos de Dados (SGBD), existem as seguintes categorias de restrições de integridade:

- a) Integridade de domínio, Integridade de vazio, Integridade de chave, Integridade referencial.**
- b) Integridade de acesso, Integridade de entrada, Integridade de saída, Integridade referencial.**
- c) Integridade de domínio, Integridade de completude, Integridade de chave, Integridade posicional.**
- d) Integridade de cardinalidade, Integridade de vazio, Integridade de autorização de acesso, Integridade associativa.**
- e) Integridade de generalização/especialização, Integridade de usuários, Integridade de chave, Integridade referencial.**

13. (ESAF/ATI/SUSEP 2006) Em um Banco de Dados Relacional:

- a) uma relação está na 1FN (primeira forma normal) se nenhum domínio contiver valores atômicos.**
- b) uma Chave Primária corresponde ao Identificador único de uma determinada relação. Em uma relação pode haver mais que uma coluna candidata a chave primária.**
- c) as colunas que irão compor as Chaves Primárias devem ser inicializadas com valores nulos.**

d) em uma tabela existirão tantas Chaves Primárias quantas forem as colunas nela existentes.

e) uma Chave Externa é formada por uma coluna de uma tabela que se referencia a uma Coluna qualquer de outra tabela. Essas colunas, na tabela destino, não aceitam valores nulos. Uma tabela destino pode ter apenas uma Chave Externa.

14. (ESAF/AFRF/SRF 2005) Com relação aos conceitos básicos de banco de dados, é correto afirmar que:

a) a chave primária é um atributo de uma tabela que, mesmo com valores nulos, identifica univocamente uma coluna.

b) o modelo relacional refere-se à visualização física e não lógica dos dados. Está relacionado ao nível conceitual interno. A teoria relacional não diz nada sobre o nível externo, preocupa-se somente com o armazenamento e manipulação dos dados executados pelo SGBD.

c) chaves estrangeiras são os elos de ligação entre as tabelas. Uma coluna definida como chave estrangeira deve ser chave primária em outra tabela.

d) um banco de dados relacional é um conjunto de arquivos seqüenciais que são acessados e modificados por operações que manipulam a álgebra relacional. Tais operações só podem ser executadas se atenderem à regra da primeira forma normal, devendo-se manipular apenas um dado de cada vez.

e) uma coluna definida como chave-estrangeira em uma tabela-destino não pode aceitar valores nulos e, essa mesma tabela-destino pode ter uma e somente uma coluna definida como chave-estrangeira.

5. GABARITOS

1	a	2	c	3	c	4	b	5	c
6	c	7	d	8	b	9	b	10	b
11	a	12	a	13	b	14	c		