



Cursos

➔ Listagem de disciplinas

Selecione uma disciplina

Aulas

- 01 Introdução a Banco de Dados
- 02 Modelo de Entidade e Relacionamento
- 03 Modelo Relacional
- 04 Transformações ER para MR
- 05 Transformações ER para MR e dicionário de dados
- 06 Normalização básica
- 07 Normalização avançada
- 08 Introdução à Linguagem SQL e Sistemas Gerenciadores de Banco de Dados
- 09 Linguagem SQL - criação, inserção e modificação de tabelas
- 10 Linguagem SQL - Consulta simples de tabelas
- 11 Linguagem SQL - Consulta avançada de tabelas
- 12 Linguagem SQL - Alteração da estrutura de tabelas e ambientes de múltiplas tabelas
- 13 Linguagem SQL - Subconsultas
- 14 Linguagem SQL - VISÕES
- 15 Linguagem SQL - STORED PROCEDURES
- 16 Linguagem SQL - Funções
- 17 Linguagem SQL - Segurança
- 18 Engenharia Reversa
- 19 Utilizando SQL em Java
- 20 Utilizando conceitos avançados de SQL em Java

⬅ Voltar 🖨 Imprimir ⬆ Topo

Sistemas de Banco de Dados

Aula 3 – Modelo Relacional

Professores autores

Nélio Alessandro Azevedo Cacho (neliocacho@ect.ufrn.br)

Xiankleber Cavalcante Benjamim (xianklebercb@gmail.com)

Apresentacao

Nesta terceira aula, iremos mostrar o projeto lógico do banco de dados. O projeto lógico pode ser feito utilizando-se diversas tecnologias. Para a nossa disciplina, iremos utilizar o Modelo Relacional e nesta aula você aprenderá alguns conceitos relacionados a esse modelo. O modelo relacional é o mais utilizado nos dias atuais e é suportado por diversas ferramentas. No nosso caso, iremos utilizar a ferramenta *MySql Workbench* para definir os elementos no modelo relacional.

Objetivo

Ao final desta aula, você será capaz de:

- conceituar o Modelo Relacional;
- definir os conceitos de Tabelas, Atributos, Domínio, Tuplas, Chave Primária e Estrangeira;
- utilizar a ferramenta *MySQL Workbench* para definir os conceitos acima do Modelo Relacional.

Modelo Relacional

Na aula anterior, falamos sobre Modelo de Entidade e Relacionamento. Vimos seus conceitos, conhecemos sua história, estudamos as entidades, os atributos, os relacionamentos e suas cardinalidades. Vimos que o Modelo ER é independente de SGDB e, portanto, deve ser o primeiro modelo gerado após a entrevista para levantamento de requisitos.

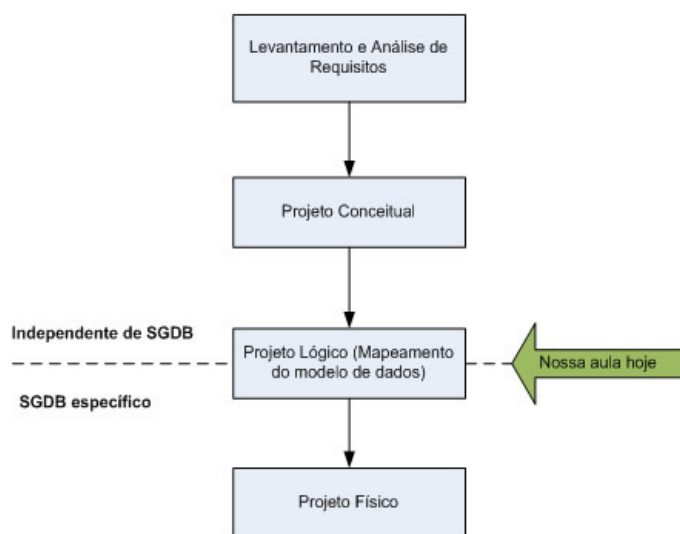


Figura 1 - Fases do Projeto de um Banco de Dados

Note que segundo a Figura 1, o projeto lógico é uma transição entre o projeto conceitual e o físico. Ou seja, estamos saindo do projeto conceitual independente de SGDB para o físico que é específico de SGDB. Na nossa disciplina, iremos utilizar como modelo específico de SGDB o modelo relacional.

O Modelo Relacional foi introduzido por Edgar Frank Codd (1970) e tornou-se um padrão para aplicações comerciais, devido a sua simplicidade e desempenho. É um modelo formal, bastante representativo e ao mesmo tempo bastante simples, foi o primeiro modelo de dados descrito teoricamente.

Um dos SGBD's precursores que implementaram esse modelo foi o System R (IBM), cuja sua história foi vista na nossa primeira aula, baseado em seus conceitos surgiram: DB2 (IBM), SQL-DS (IBM), Oracle, Informix, Ingres, Sybase, entre outros.

O Modelo Relacional representa os dados num Banco de Dados como uma coleção de relações e seus relacionamentos. Cada relação contém um nome e um conjunto de atributos com seus respectivos nomes. Informalmente, as relações do Modelo Relacional são também chamadas de tabelas pela maioria dos desenvolvedores. Como a nossa ferramenta MySQL Workbench usa o nome "tabela" ao invés de "relação", iremos de agora em diante chamar relações de tabelas.

Tabela

Toda a informação de um banco de dados relacional é armazenada em tabelas, que, na linguagem do modelo relacional, também são chamadas de relações (Batisti, 2010).

Por exemplo, posso ter uma Tabela "Empregado", onde seriam armazenadas informações sobre os diversos empregados.

No entanto, como posso armazenar as informações em uma tabela? Para responder tal pergunta, vamos aprender o conceito de atributos.

Atributos

Atributos são todas as informações que existem em uma tabela. Essas informações são chamadas informalmente de campos.

Exemplo: Nome, CPF, Rua, Bairro, Telefones, CEP, Data de Nascimento etc.

Domínio

Todo atributo para armazenar as informações de uma tabela deve ter um domínio definido. O domínio representa todos os valores possíveis que um atributo pode receber. Por exemplo, o atributo Telefone pode receber um conjunto de número com oito dígitos. Por outro lado, o atributo Nome pode receber um conjunto de cadeias de caracteres que representa o nome de uma pessoa. Desse modo, o domínio de um atributo define qual o tipo de dado e o formato que o dado pode ser armazenado por aquele atributo. Por exemplo, o formato do atributo Data de Nascimento é "dd/mm/ano". O formato do atributo CEP é "nnnnn-nnn". Assim, o formato descreve como o dado será exibido para o usuário do sistema.

Tuplas

As tuplas representam os valores de uma tabela. A Figura 2 mostra uma tabela Empregado preenchida com valores hipotéticos. Note que as colunas da tabela representam os atributos, enquanto as linhas representam as tuplas. Se uma tabela não tiver tuplas, ela estará vazia, ou seja, sem dados. Desse modo, quando efetuarmos uma busca no Google, recebemos como resposta as tuplas do banco de dados do Google que estão relacionadas de alguma forma com o texto procurado. Informalmente, as tuplas são também chamadas de registros pelos desenvolvedores.

Matricula	Nome	Sexo	Endereco	Telefone
1	Nelio	M	Rua das Na...	8888-1555
2	Jose	M	Rua das ca...	8888-9999
3	Maria	F	Rua das Ala...	9999-7444

Figura 2 - Os atributos e as tuplas de uma tabela Empregado

Depois de aprender o significado dos conceitos Tabela, Atributo, Domínio e Tuplas, é importante agora aprender como utilizar a ferramenta *MySQL Workbench* para definir tais conceitos no Modelo Relacional.

Utilizando a Ferramenta *MySQL Workbench* para definir o Modelo Relacional

Antes de tudo, vamos abrir a ferramenta '*MySQL Workbench*' que encontra-se no menu iniciar do Windows, conforme a Figura 3. Caso você use Linux, a ferramenta provavelmente será instalada no menu de aplicações.

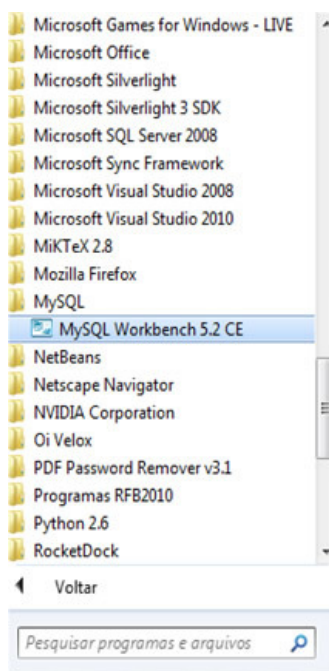


Figura 3 - Menu Iniciar *MySQL Workbench* no Windows

Na Figura 4, temos a tela inicial do '*MySQL Workbench*'. Para começar, vamos escolher a opção '*Create New Err Model*'. Escolhendo essa opção, avançamos para uma próxima tela, na qual temos a opção '*Add Diagram*'. Essa tela pode ser visualizada pela Figura 5. Clicando duas vezes em '*Add Diagram*', abrimos nossa área para criarmos o Modelo de Relacional.

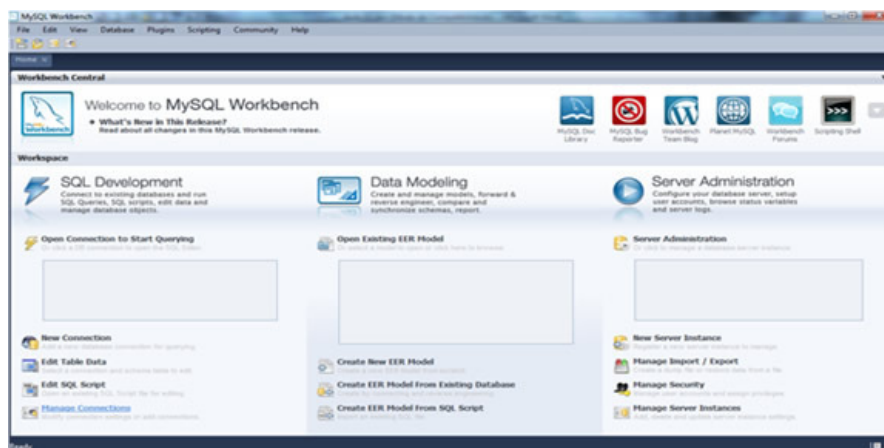


Figura 4 - Tela de abertura do *MySQL Workbench*

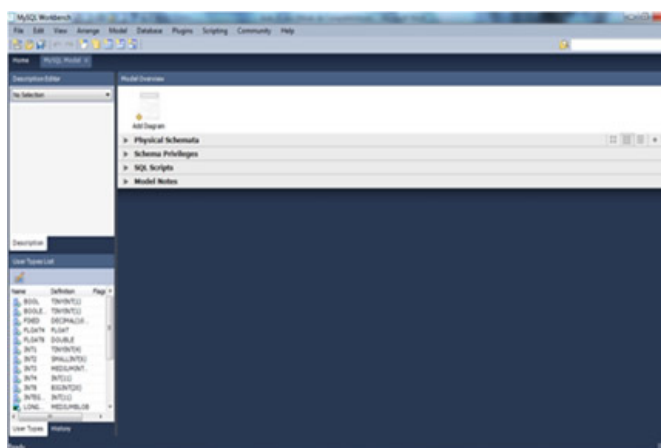


Figura 5 -Ambiente para criação dos diagramas no *MySQL Workbench*

Pronto, agora você já pode definir os elementos do Modelo Relacional utilizando o '*MySQL Workbench*'. Para exemplificar, vamos mostrar como criar uma tabela e definir seus atributos. Para começar, vamos definir a tabela *Empregado* usando a ferramenta '*MySQL Workbench*'. Na Figura 6, temos o ambiente da ferramenta onde iremos criar nosso Modelo Relacional. O modelo é criado na parte branca que ocupa a maior parte do lado inferior direito da ferramenta.

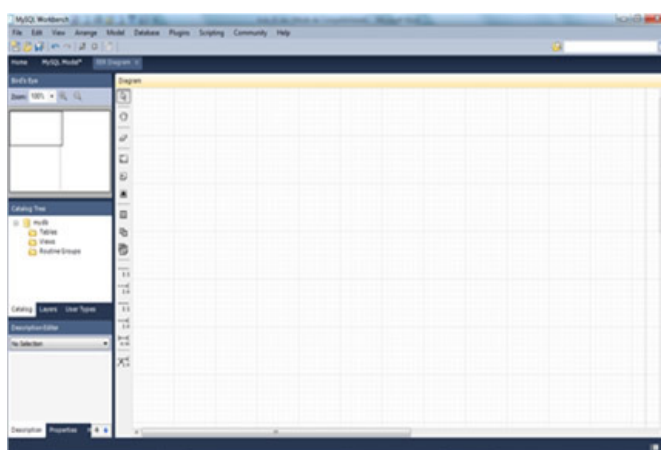


Figura 6 - Ambiente de trabalho para desenvolver o Modelo Relacional no *MySQL Workbench*

E agora, como criamos essa nossa tabela? Vejamos o passo a passo a seguir.

1. Na barra mostrada pela Figura 7, podemos ver vários ícones e um número ao lado de cada ícone. Para criar uma tabela, você deve clicar no ícone que possui o número 7 e logo em seguida clicar na parte branca situada logo a direita da barra mostrada na Figura 7.

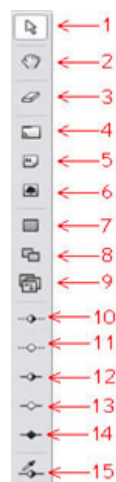


Figura 7 - Barra de Ferramentas do *MySQL Workbench*

2. Ao clicar com o botão direito do mouse em cima da tabela criada, você irá ver a opção *'Edit Table'*. Quando clicar nessa opção, vai aparecer, na parte inferior da tela, uma aba onde existe a opção *'Table'*. Nessa aba, você irá ver um campo de texto com nome "Name" onde colocamos o nome da tabela. Note que "Table" em inglês equivale à tabela em português. Para o nosso exemplo, você deve fornecer o nome Empregado. Ao final dessa operação, você deve obter algo parecido com a Figura 8.

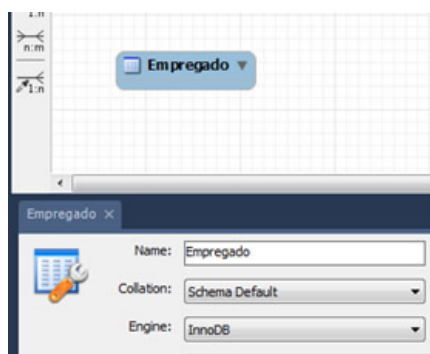


Figura 8 - Exemplo de criação da Tabela no *MySQL Workbench*

3. Agora, devemos definir os atributos da tabela. Para isso, devemos clicar na aba inferior chamada *'Columns'*. Você deve informar o nome do atributo e o tipo em cada linha da tabela. Note que ao clicar na aba *'Columns'* o *MySQL Workbench* cria automaticamente um atributo com nome "idEmpregado" com tipo INT.

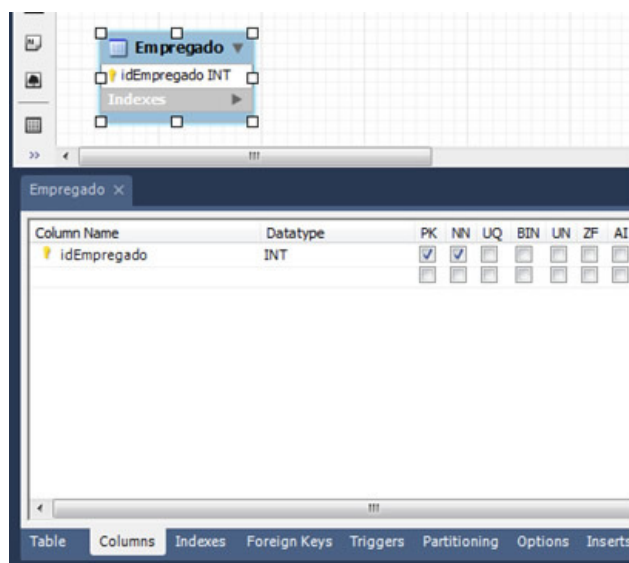


Figura 9 - Exemplo de definição de Atributos no *MySQL Workbench*

Para criar um atributo com nome e tipo desejado, você deve clicar duas vezes em cima da linha onde está o nome do atributo "idEmpregado". Quando o cursor ficar como o mostrado na Figura 10, você pode digitar o nome do atributo desejado.

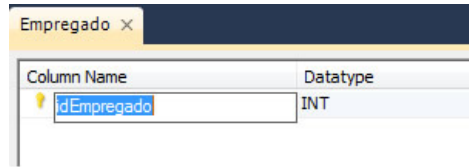


Figura 10 - Exemplo de definição de nome de Atributos no *MySQL Workbench*.

Para o nosso exemplo, você deve digitar "Matricula". Para criar outro atributo, você precisa apenas clicar duas vezes na linha abaixo do atributo criado. Tente repetir essa operação para criar os atributos: Nome, Sexo, Endereço e Telefone.

Depois de definir o nome dos atributos, você precisa definir os tipos de cada um. Para definir o tipo de cada atributo, você deve clicar na coluna "Datatype" de cada atributo. Ao clicar, irá aparecer uma lista com várias opções de tipos suportadas pelo *MySQL*, como mostrado pela Figura 11.

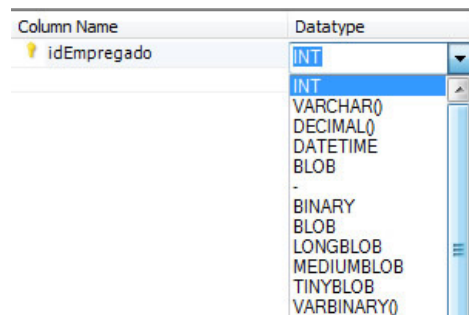


Figura 11 - Exemplo de definição de tipo de Atributos no *MySQL Workbench*

Você deve definir os tipos de modo que o seu modelo fique parecido como o mostrado na Figura 12. Note que os tipos dos atributos suportados variam entre os SGBDs. No caso do *MySQL*, os principais tipos suportados são:

Tabela 1 - Descrição dos tipos de dados suportada pelo *MySQL*

Tipo	Descrição
VARCHAR	Valores no campo VARCHAR são strings de tamanho variável. Você pode declarar um campo VARCHAR para ter qualquer tamanho entre 1 e 255, assim como para campo CHAR. No entanto, diferente de CHAR, valores VARCHAR são armazenados usando apenas quantos caracteres forem necessários, mais 1 byte para gravar o tamanho.
INT	Valores inteiros de -2147483648 a 2147483647.
DECIMAL	O tipo DECIMAL é usado por valores para os quais é importante preservar a exatidão como, por exemplo, dados monetários.
DATA	O tipo DATA é usado quando se necessita apenas do valor da data, sem a parte da hora. MySQL recupera e mostra valores do tipo DATA no formato 'ano-mm-dd'

Fonte [Manual de Referência do *MySQL* 4.1]

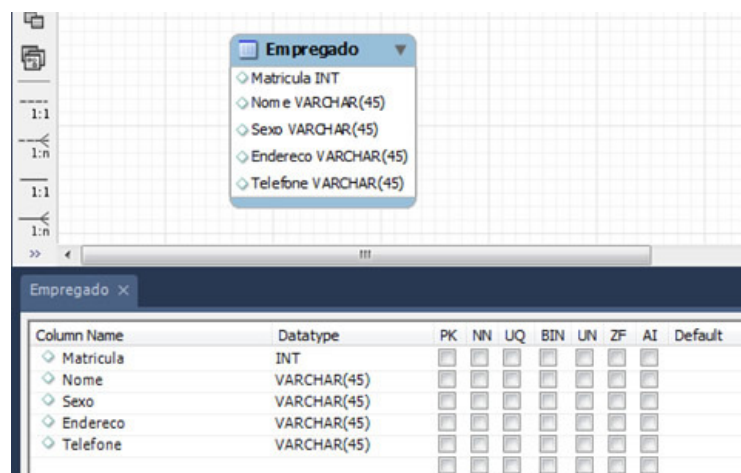


Figura 12 – Exemplo de definição de uma tabela no MySQL Workbench

Pratique 01

1. Repita os passos acima para definir as tabelas abaixo.
 - a. Tabela Departamento com atributos Nome, Número, Localização.
 - b. Tabela Dependente com atributos Nome, sexo, DataNascimento e Parentesco.

Um dos grandes desafios em se projetar um Banco de Dados com sucesso é a correta determinação das Tabelas que existirão no Banco de Dados, bem como dos Atributos de cada Tabela. (Batisti, 2010).

Outros conceitos do Modelo Relacional

Neste momento da aula, você irá aprender outros dois conceitos que são importantes para manter a integridade do Modelo Relacional, sendo eles: chaves primária e estrangeiras.

Chave primária

Na aula passada, vimos que entidades podem possuir um atributo Chave que é usado para garantir a distinção entre as ocorrências das entidades. No Modelo Relacional, o conceito de chave primária equivale ao conceito de atributo chave do modelo ER. Um atributo que for chave primária em uma tabela não pode ter tuplas (registro) com o mesmo valor para aquele atributo e também não pode conter nenhuma tupla sem valor. Um registro que não possui valor é chamado de registro **nulo** (do inglês *null*). As chaves primárias identificam de maneira única cada registro/tupla de uma tabela. Veja exemplos de campos que podem ser definidos como chaves primárias.

- Campo CPF em uma tabela de cadastro de clientes.
- Campo CNPJ em uma tabela de cadastro de fornecedores.
- Matrícula do aluno em uma tabela de cadastro de alunos.
- Código da Peça em uma tabela de cadastro de peças.
- Matrícula do funcionário em uma tabela de cadastro de funcionários.
- Número do pedido em uma tabela de cadastro de pedidos

Na Figura 13, temos uma tabela com uma chave primária no atributo Matricula. Bom, até agora aprendemos a criar uma tabela e inserir atributos, mas, como vamos inserir uma chave primária?



Figura 13 –Tabela com chave primária

Para inserirmos a chave primária, temos que marcá-la, na Figura 14 há vários quadradinhos ao lado de cada atributo com algumas opções, vamos marcar a opção PK, que é a nossa chave primária. O termo PK vem do inglês "Primary Key", que em português quer dizer chave primaria.

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI
Matricula	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Nome	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Sexo	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Endereco	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Telefone	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figura 14 –Tabela com chave primária

Chave estrangeira

Uma coisa importante no Modelo Relacional é a integridade das informações armazenadas. Para garantir essa integridade, é necessário que as informações em uma tabela estejam relacionadas com outras informações em outras tabelas. Por exemplo, a Figura 15 mostra a tabela Dependente que possui o atributo Matricula. Esse atributo refere-se à matrícula do Empregado pela qual o dependente depende. Ou seja, para ser íntegro, toda a matrícula que for cadastrada na tabela dependente deve estar cadastrada na tabela Empregado. Isso irá garantir que um dependente depende de um empregado cadastrado.

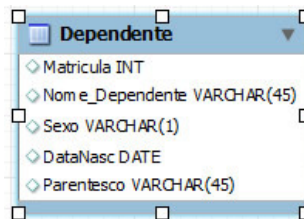


Figura 15 – Tabela Dependente sem relação de integridade com a Tabela Empregado

No entanto, como garantir tal regra de integridade? A resposta para essa pergunta chama-se chave estrangeira. Uma chave estrangeira estabelece uma relação de integridade referencial com uma chave primária de outra tabela. Isso garante que os valores de um atributo (chave estrangeira) em uma tupla de Dependente só serão aceitos se o mesmo valor existir em outro atributo (chave primária) da tabela Empregado.

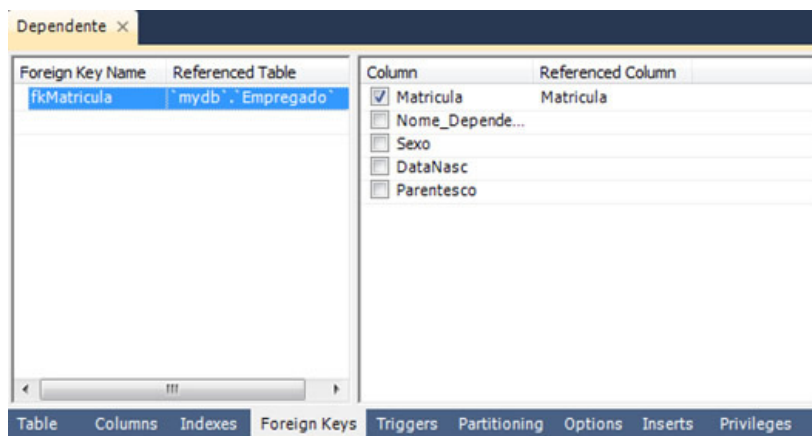
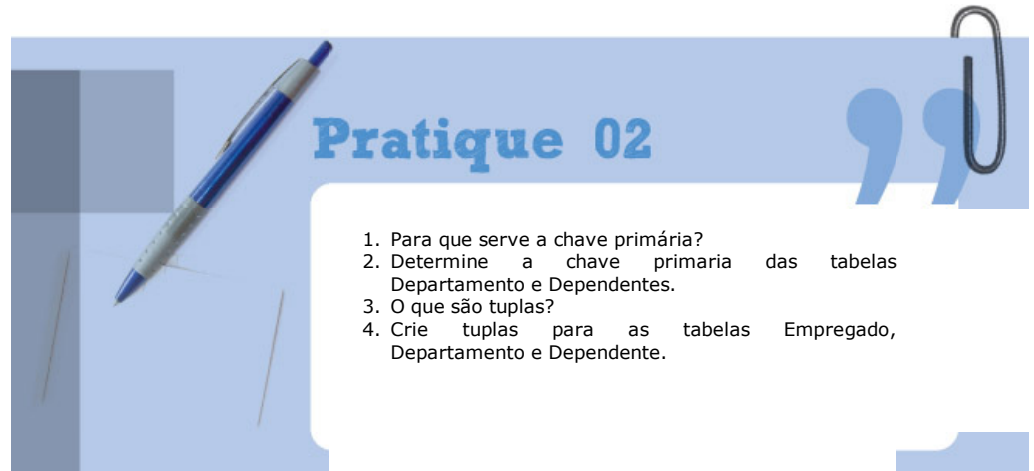


Figura 16 – Definindo chave estrangeira no MySQL Workbench

Bom, vamos aprender agora como inserimos nossa chave estrangeira utilizando a ferramenta 'MySQL Workbench'. Para inserirmos a chave estrangeira, temos que fazer duas coisas. Primeiro, marcar nos quadradinhos ao lado do atributo Matricula na tabela Dependente a opção NN (not null). O segundo passo é ir à aba 'Foreign Keys' e definir o nome da chave estrangeira no campo "Foreign Key name", veja o exemplo na Figura 16. Depois disso, você deve informar qual tabela ("Referenced Table") essa chave estrangeira irá se referenciar. No nosso caso, a tabela referenciada é a tabela Empregado. Para finalizar, você deve informar qual atributo da tabela Empregado será utilizada para controlar a integridade da tabela dependente. Nesse caso, você deve marcar o atributo Matricula.

Depois do exemplo da Figura 16, a tabela Dependente só aceitará matrículas que estiverem cadastradas na tabela Empregado. Note que esse controle é feito automaticamente pelo SGBD, uma vez que você utilizou o recurso de chave estrangeira. Se a chave estrangeira não fosse utilizada, você teria que implementar tal controle no próprio sistema. Na maioria dos casos, isso não é recomendado.

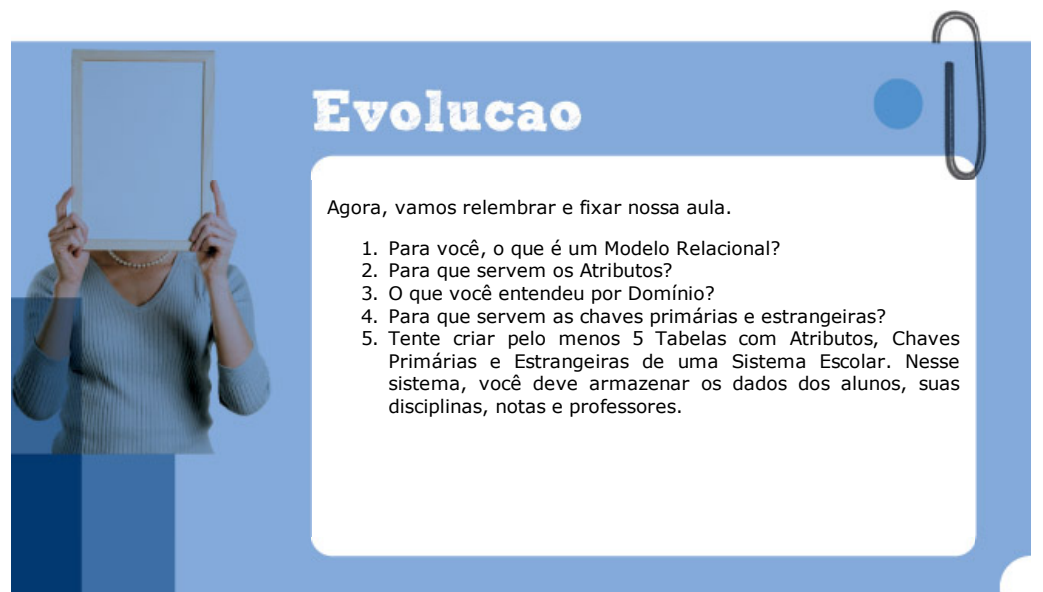


Pratique 02

1. Para que serve a chave primária?
2. Determine a chave primária das tabelas Departamento e Dependentes.
3. O que são tuplas?
4. Crie tuplas para as tabelas Empregado, Departamento e Dependente.

Resumo

Nesta terceira aula, vimos o assunto Modelo Relacional (MR), conhecemos um pouco mais sobre a sua importância e história, aprendemos seus conceitos e vimos vários exemplos de entidade, atributos, tipos de dados e registros. Vimos também a importância das chaves primária e estrangeira, bem como exemplos práticos de utilização da ferramenta 'MySQL Workbench'.



Evolucao

Agora, vamos relembrar e fixar nossa aula.

1. Para você, o que é um Modelo Relacional?
2. Para que servem os Atributos?
3. O que você entendeu por Domínio?
4. Para que servem as chaves primárias e estrangeiras?
5. Tente criar pelo menos 5 Tabelas com Atributos, Chaves Primárias e Estrangeiras de uma Sistema Escolar. Nesse sistema, você deve armazenar os dados dos alunos, suas disciplinas, notas e professores.

Referencias

BATTISTI, Júlio Cesar Fabris Linha de Código. **Conceitos**. Disponível em: <<http://www.linhadecodigo.com.br/artigo/109/o-modelo-relacional-de-dadosparte-i.aspx>>. Acesso em: 23 jul. 2010.

DATE, C. J. **Introduction to database systems**. 7th ed. Boston: Addison Wesley Longman, 1999.

DATE, Christopher J. **Introdução a sistemas de banco de dados**. Rio de Janeiro: Campus, 2000.

FILHO, Mário. **Modelo lógico de dados**. Disponível em: <http://www.3dinformatica.com.br/fama/downloads/5_periodo/Encontro%2007%20-%20MR%2001.ppt>. Acesso em: 16 julho. 2010.

HEUSER, C. A. **Projeto de banco de dados**. 6. ed. Porto Alegre: Editora Bookman, 2009.

MANUAL de Referência do MySQL 4.1 Disponível em: <<http://dev.mysql.com/doc/refman/4.1/pt/date-and-time-types.html>>. Acesso em: 18 ago. 2010.

WIKIPÉDIA. **Chave primária** Disponível em: <http://pt.wikipedia.org/wiki/Chave_prim%C3%A1ria>. Acesso em: 16 jul. 2010.

_____. **Chave estrangeira**. Disponível em: <http://pt.wikipedia.org/wiki/Chave_estrangeira>. Acesso em: 16 jul. 2010.



Voltar



Imprimir



Topo