

# INFORMÁTICA PARA INTERNET

---

## Modelagem de banco de dados

---

### Introdução

Neste material, vamos aprofundar nosso conhecimento em banco de dados. Agora que você já compreende algumas regras, conceitos e estrutura dos comandos SQL, vamos aprender a manipular de forma mais contundente o sistema gerenciador de banco de dados, também conhecido como SGBD.

Você irá aprender a modelagem conceitual de um banco de dados, assim como irá estudar técnicas de levantamento de dados e requisitos funcionais e não funcionais.

Além disso, irá aprender sobre diagrama UML e sua aplicação, conhecendo alguns dos principais diagramas utilizados pelas equipes de desenvolvimento de *software* e saberá a importância da utilização deste diagrama.

Neste material, você vai estudar os modelos relacionais embasados na teoria dos conjuntos, que fazem o relacionamento entre as tabelas e deixam os dados acessíveis e possíveis de serem consultados de diferentes formas. Ainda, aprenderá a fazer a instalação e a configuração do banco de dados MySQL pelo XAMPP e pelo MySQL Workbench.

(#topo)

## Modelagem conceitual

---

O modelo conceitual do banco de dados é uma representação simples. Trata-se de uma representação gráfica das estruturas (tabelas) do banco de dados em que é possível identificar todas as relações e restrições e suas demais características.

O desenvolvimento deste modelo é considerado uma das primeiras e principais etapas no projeto de desenvolvimento do banco de dados. Os projetos de banco de dados mantêm seu foco na maneira como sua estrutura será utilizada para armazenar e gerenciar os dados do usuário final.

A modelagem de dados começa com a compreensão do ambiente real, com escopo e objetivos bem-definidos. Se realizado de maneira adequada, o modelo de dados final é equivalente a uma “planta”, com todas as instruções para a construção de um banco de dados que atenda às necessidades dos usuários

finais.

O **modelo conceitual de dados mais utilizado atualmente**, devido principalmente à simplicidade e à eficiência, é o modelo entidade-relacionamento, também chamado de **diagrama entidade-relacionamento (DER)**.

Para construir um banco de dados sem erros, precisamos aprender a utilizar técnicas pertinentes a sua construção. A partir de agora, você vai conhecer e estudar as principais técnicas, estratégias e definições para a construção do banco de dados.

## Técnicas de levantamento de dados

O levantamento de dados é uma forma de mapeamento e o principal objetivo é permitir conhecer as principais e reais necessidades do cliente. Em grande parte dos casos, o cliente não sabe expressar de forma clara as reais necessidades que precisam ser atendidas com a construção e a utilização de *software* com banco de dados.

Para realizar esses levantamentos adequadamente, algumas técnicas devem ser aplicadas.

### **Observação direta, ou observação pessoal**

Esta técnica permite vivenciar o dia a dia da empresa. Por meio dela, pode-se obter a confirmação de informações recebidas, como erros de procedimentos. Este é um elemento importante quando outras técnicas já tiverem sido aplicadas, como a entrevista, por exemplo.

### **Questionário**

Este é um instrumento normalmente preparado em formulário para levantar determinadas informações. O questionário pode ser aberto ou fechado: o fechado apresenta questões e suas possíveis respostas; já o aberto permite que cada participante responda às questões de acordo com o seu entendimento. O questionário deve ser elaborado cuidadosamente, principalmente quando se tratar do tipo aberto, para que as respostas não sejam pobres, como simplesmente “sim” e “não”.

### **Entrevista**

Assim como o questionário, a entrevista também apresenta algumas questões. A grande diferença é que os questionamentos, neste caso, podem ser modificados durante a aplicação do instrumento.

### **Análise da documentação**

Esta técnica consiste na análise dos documentos já existentes que a empresa utiliza para organizar a sua rotina, como cadastro de clientes.

As informações obtidas a partir da aplicação dessas técnicas serão norteadoras para o desenvolvimento dos requisitos e das demais características do projeto de banco de dados.

É muito comum que precise ser realizada mais de uma técnica para compreender o que precisa ser modelado. Não existe uma fórmula pronta e cada levantamento de dados poderá englobar várias técnicas.

## Levantamento e especificação de requisitos

A partir do levantamento de dados, estamos aptos para realizar especificações e requisitos a que o *software* deve atender.

Todo projeto de *software* tem um conjunto de requisitos, os quais são determinados pelas necessidades e pelas expectativas dos usuários que efetivamente o utilizarão. Além disso, os requisitos devem estar relacionados ao atendimento dos objetivos dos negócios da empresa.

Um bom requisito deve endereçar uma necessidade direta ou indireta dos futuros usuários do sistema a ser desenvolvido.

A prática recomendada é sempre documentar, organizar e disponibilizar os requisitos a todos os envolvidos no projeto, garantindo que seu entendimento seja compartilhado entre todos, ou seja, clientes e equipe responsável pelo projeto.

Existe uma variedade de tipos de requisitos, que deverão estar relacionados a funcionalidades esperadas para o sistema e também ao desempenho, à segurança e à confiabilidade de informações. Quando tratamos de projeto de desenvolvimento de *softwares*, os requisitos dividem-se em requisitos funcionais e não funcionais.

Há várias maneiras de documentar e disponibilizar esses requisitos. Uma das ferramentas para essa documentação é o diagrama ER, que monta as entidades e os relacionamentos entre elas. Outra ferramenta possível são os diagramas UML (*unified modeling language*).

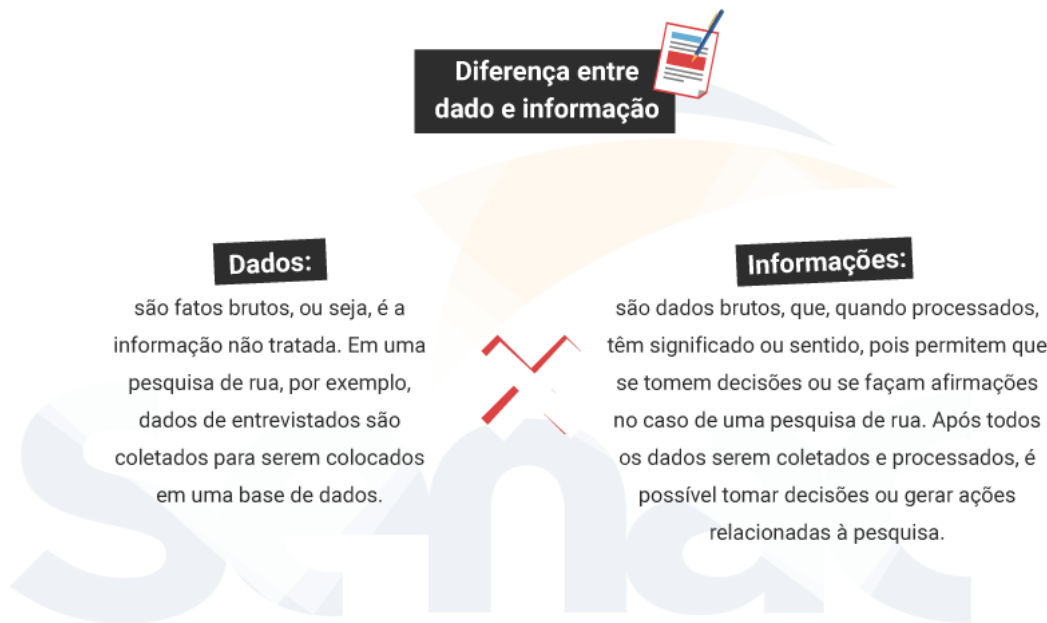
(#topo)

## Banco de dados – Modelagem de dados por meio de ER

---

Entidade-relacionamento (ER) é um modelo de dados conceitual de alto nível cujos conceitos foram projetados para se aproximarem o máximo possível da visão que o usuário tem dos dados. Não há uma preocupação em representar o modo como esses dados estarão realmente armazenados.

Existe uma diferença entre dados e informações e percebê-la é fundamental para compreender como o projeto de banco de dados será orientado.



#### Diferença entre dado e informação

**Dados:** são fatos brutos, ou seja, é a informação não tratada. Em uma pesquisa de rua, por exemplo, dados de entrevistados são coletados para serem colocados em uma base de dados.

**Informações:** são dados brutos, que, quando processados, têm significado ou sentido, pois permitem que se tomem decisões ou se façam afirmações no caso de uma pesquisa de rua. Após todos os dados serem coletados e processados, é possível tomar decisões ou gerar ações relacionadas à pesquisa.

## Modelo ER

O modelo ER baseia-se na percepção do mundo real e consiste na **coleção de objetos denominados entidades e suas relações**.

A abordagem ER foi criada em 1976, por Peter Chen, e tem como ideologia básica a percepção do mundo real. Esta abordagem é representada por meio de uma coleção de objetos chamados de entidade e dos relacionamentos entre esses objetos. Ela pode ser considerada um padrão para modelagem conceitual. Isso forneceu uma nova e importante percepção dos conceitos de modelos de dados.

Por esse motivo, o modelo ER é utilizado em projetos para modelar os conceitos do banco de dados de forma independente do SGBD.

## Diagrama ER

O DER é a **representação gráfica do modelo ER**. O DER é uma forma de visualizar as informações, já que o modelo pode ficar abstrato demais para auxiliar no desenvolvimento do sistema. Assim, quando se está modelando um banco, o mais comum é criar a sua representação gráfica.



Figura 1 – Diagrama ER

Imagem com dois retângulos que significam as entidades “cliente” e “produto”. No meio dos retângulos, há um losango, que representa a ação.

O modelo ER e o DER são importantes ferramentas utilizadas durante o desenvolvimento de sistemas, geralmente naqueles mais complexos e difíceis de visualizar.

Quando a modelagem é executada de forma correta durante o desenvolvimento da base de dados, evita-se a necessidade de alterações para corrigir erros provenientes de falhas durante a análise do projeto.

O DER baseia-se em uma percepção de um mundo real que consiste em uma coleção de objetos básicos, chamados de entidade e relacionamento.

## Entidade

Uma entidade é uma coisa ou um objeto no mundo real que é distinguível de outra coisa ou outro objeto. As entidades podem ser objetos físicos, como um cliente ou um produto, mas também abstrações, como rotas de voo ou apresentações musicais. Cada cliente, por exemplo, é representado por uma entidade, mas produtos também podem ser entidades.

A figura a seguir representa graficamente uma entidade:



Figura 2 – Representação de entidade

Temos a estrutura de duas entidades. A entidade é representada por um retângulo, e o seu nome deve ser escrito dentro dela. No caso, temos a entidade cliente e a entidade produto.

## Atributos

As entidades são descritas em um banco de dados por um conjunto de atributos. Por sua vez, um atributo é uma característica de uma entidade. Os atributos **nome** e **telefone** podem ser atribuídos a uma entidade denominada **cliente**. O atributo pode ser considerado uma característica, uma identificação, um endereço, um nome ou outro.

Há vários tipos: simples, compostos, multivalorados e determinantes (identificadores). Veja a seguir as características de cada um deles.

## Simple

Não tem característica especial, como nome, na entidade cliente.

## Identificador ou determinante

Identifica de forma única uma entidade. Isso significa que não pode haver dados repetidos. É o caso do CPF ou do CNPJ de uma empresa, por exemplo.

## Composto

O seu conteúdo é formado por vários itens menores, assim como um endereço é formado por rua, CEP e bairro.

## Multivalorado

O seu conteúdo é formado por mais de um valor, como telefone.



Figura 3 – Cliente com atributos

Imagem com um retângulo com a entidade “cliente” e quatro atributos, que são: nome endereço, CPF e telefone. O campo CPF está marcado como chave primária e o campo endereço contém novos atributos, que descrevem o endereço (rua, CEP e bairro)

## Relacionamentos

Uma relação é uma associação entre várias entidades e é representada por uma ação, como a ação de “comprar” associa o cliente ao produto. O símbolo que representa o relacionamento é um losango, com seu nome escrito no seu interior.



Figura 4 – Ação entre as entidades

## Cardinalidade

Um conceito muito relevante e que não se pode deixar de fora quando se trata de relacionamentos é a cardinalidade, que determina o número de ocorrências em um relacionamento.

Para determinar a cardinalidade, deve-se fazer a pergunta relativa ao relacionamento em ambas as direções.

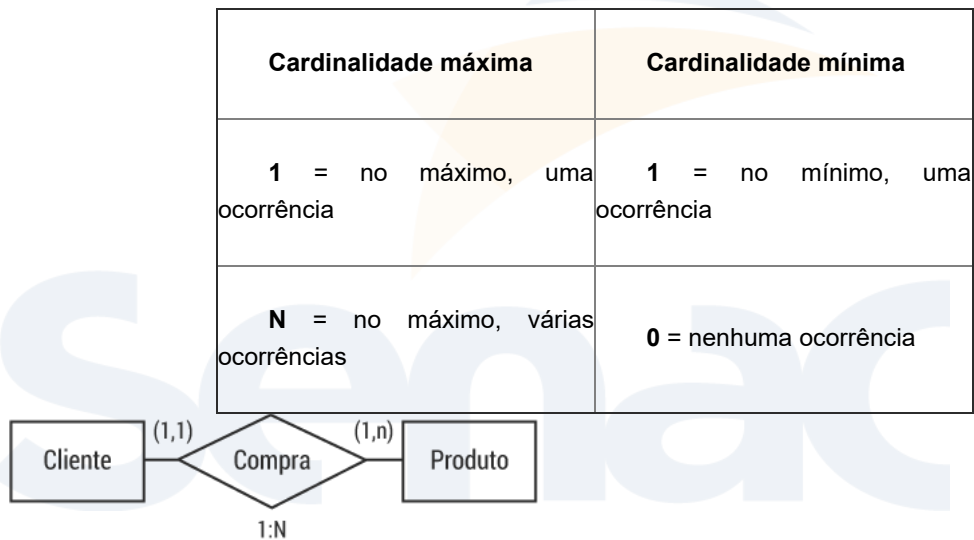


Figura 5 – Tipos de cardinalidades

Dois retângulos que representam as entidades e um losango que representa a ação. Os retângulos estão ligados ao losango pelas extremidades e nestas ligações estão expressas as cardinalidades.

- ◆ Um cliente compra quantos produtos? No mínimo, 1; e, no máximo, N.
- ◆ Um produto pode ser comprado por quantos clientes? No mínimo, 1; e, no máximo, 1.
- ◆ Somando-se as cardinalidades, definimos assim o resultado do relacionamento: 1:N.  
Há diferentes tipos de relacionamentos existentes entre as entidades.
- ◆ **Um para um:** uma entidade A se relaciona com uma entidade B. Observe a sua aplicação:



Figura 6 – Cardinalidade um para um

Dois retângulos que representam as entidades e um losango que representa a ação. Os retângulos estão ligados ao losango pelas extremidades e nestas ligações estão expressas as cardinalidades.

- ◆ **Um para muitos, ou um para vários:** uma entidade A se relaciona com uma ou mais entidades B. Observe a sua aplicação:



Figura 07 – Cardinalidade um para muitos.

Dois retângulos que representam as entidades e um losango que representa a ação. Os retângulos estão ligados ao losango pelas extremidades e nestas ligações estão expressas as cardinalidades.

A interpretação do diagrama é a seguinte: um cliente pode comprar vários produtos, porém um produto só pode ser comprado por um cliente.

◆ **Muitos para muitos, ou vários para vários:** muitas (N) entidades A se relacionam com muitas entidades B (M). Observe a aplicação:



Figura 8 – Cardinalidade muito para muitos

Dois retângulos que representam as entidades e um losango que representa a ação. Os retângulos estão ligados ao losango pelas extremidades e nestas ligações estão expressas as cardinalidades.

Interpreta-se assim o diagrama: um cliente pode comprar vários produtos, e vários produtos podem ser comprados por um cliente.

Veja exemplo completo de montagem de um DER nas videoaulas desta unidade curricular (UC).

## Dicionário de dados

O dicionário de dados fornece uma descrição detalhada de todas as tabelas encontradas no banco de dados criado pelo usuário ou pelo projetista. Logo, contém no mínimo todos os nomes e as características dos atributos de cada uma das tabelas do sistema.

O dicionário tem por finalidade garantir que todos os membros das equipes de projeto e da implementação do banco de dados utilizem os mesmos nomes e as mesmas características de tabelas e atributos.

Existe uma variedade de modelos de dicionários de dados, como podemos ver na figura a seguir:



Figura 9 – Entidade e atributos

Um retângulo que expressa a entidade e três atributos contidos na entidade: nome, endereço e CPF.

Esta entidade será utilizada como referência para a construção do dicionário de dados, representado na tabela a seguir.



Nome da tabela	Nome do atributo	Conteúdo	Tipo	Formato	Faixa	Necessário
Cliente	Nome	Nome do cliente	VARCHAR(20)	Xxxxxx		S
	Endereço	Endereço do cliente	VARCHAR(20)	Xxxxxx		
	CPF	CPF do cliente	CHAR(14)	999.999.999-99		S

Um índice é uma disposição ordenada utilizada para acessar logicamente linhas e colunas de uma tabela. Os índices no ambiente de banco de dados relacionais são estruturas que localizam rapidamente um item de acesso.

De um ponto de vista formal, um índice é uma disposição ordenada de chaves e ponteiros. Cada chave aponta para a localização dos dados identificados por ela.

Os SGBDs utilizam índices para várias finalidades, como recuperar dados de maneira mais rápida, ordenados por atributos específicos. A criação de um índice de sobrenomes de clientes, por exemplo, permitirá a recuperação alfabética dos dados dos clientes pelo seu sobrenome.

Os índices têm um papel importante na implementação das chaves primárias. Ao se definir a chave primária de uma tabela, o SGBD cria automaticamente um índice exclusivo para a sua coluna.

## Diagramas UML

A análise de requisitos, modelagem de projetos, implementação e implantação tem uma necessidade de notação padrão para o uso nas atividades. Antigamente, era comum utilizar uma notação para análise fazendo o uso de digramas de fluxo, outra para projeto estruturado, que utilizava diagramas para expressar a estrutura e a comunicação de todos os módulos e, ainda, uma última para a implantação. Havia um grande risco de perda de detalhes entre as passagens de notações, pois algumas vezes os detalhes não podiam ser expressos em outra notação e assim surgiam adaptações que produziam digramas diferentes ao fim do ciclo. Além da perda de informação, ainda poderia haver alteração das informações, levando o resultado final a ficar diferente do esperado.

A linguagem UML veio para resolver este problema, pois ela pode e deve ser usada em todas as fases de elaboração do sistema, desde a análise até a implantação final do sistema.

A linguagem UML tem vários diagramas que são utilizados tanto pelos analistas de sistemas quanto por desenvolvedores, como os diagramas de classes, os diagramas de sequências, os diagramas de atividades, os diagramas de colaboração, entre outros. Agora, a notação é idêntica e percorre todas as fases. Não há mais a necessidade de serem feitos os mapeamentos de um diagrama de análise para outro diagrama que compõe o projeto, o que gera consistência e assegura que não haverá perda de informação nem adaptações.

Designer/Diagramador: caixa de destaque com ícone de importante

Esta linguagem foi proposta para uma notação para modelagem de sistemas orientados a objeto, mas pode ser perfeitamente utilizada nos projetos de banco de dados. Os projetos de banco de dados compõem uma fase de extrema importância do desenvolvimento de qualquer sistema de informação, pois todos os dados serão salvos (armazenados) no banco de dados.

Esta linguagem tem vários digramas e o **digrama de classe** é o utilizado para representar o projeto conceitual de um banco de dados. Para evitar os problemas citados anteriormente, é necessário o uso de uma notação.

Embora o DER seja muito usado para o projeto conceitual de um banco de dados, é importante o uso de uma notação padrão para se evitar os problemas já mencionados. É muito comum que as equipes façam a modelagem conceitual de banco de dados com o DER e depois façam a modelagem de classes construindo um diagrama de classes de negócio que não tem nada a ver com o diagrama de relacionamentos e entidades.

## A abordagem de projeto de banco de dados com a linguagem UML

Agora será introduzida uma abordagem para a construção do modelo conceitual de banco de dados com o uso da linguagem UML, o que garantirá que os riscos de perda e distorções de informações entre as passagens de diferentes notações sejam sanados.

Isto será realizado partindo de outros diagramas da UML, como os diagramas de casos de uso, os diagramas de sequência, os diagramas de máquina de estados e o diagrama de atividades. Para se chegar ao diagrama de classes, os requisitos de negócio, requisitos funcionais, podem ser coletados junto ao usuário com o uso dos diagramas da UML citados. Todos estes diagramas se completam e geram informações que podem ser consultadas e armazenadas em um banco de dados e, portanto, precisam ser representadas no diagrama de classes que servirá para o projeto conceitual do banco de dados.

A principal contribuição da linguagem UML foi conseguir reunir analistas de sistemas, administradores de dados, projetistas de banco de dados e desenvolvedores. Eles conseguem trabalhar tendo por base uma notação comum e assim cada um contribui de alguma forma, ajudando a elaborar e a construir um projeto de banco de dados mais fidedigno e próximo da realidade, sem deixar de atender nenhum dos requisitos que compreendem o negócio. Como a linguagem UML suporta em diversas dimensões de modelagem (comportamental, estrutural, dinâmica e estática), a saída é um projeto de banco de dados mais completo do que se teria se apenas fosse considerada a dimensão estática.

Todos os diagramas citados devem ser lidos, interpretados e compreendidos para se construir o diagrama de classes que será usado para o projeto conceitual do banco de dados.

O objetivo é fazer a identificação dos elementos de informação que são citados nestes diagramas e que representam informação relevante ao negócio, que devem, portanto, ser adicionadas ao diagrama de classes do sistema. Logo, todo e qualquer elemento de informação que representa algo relevante ao negócio deve constar no diagrama de classes.

### Diagrama de casos de uso

O diagrama de casos de uso representa as interações funcionais entre os atores e o sistema. Cada cenário ou fluxo é uma sequência de passos que descrevem a forma como os atores interagem com o sistema a fim de obter algo de valor para o negócio (uma NFe emitida, uma matrícula efetuada, um cálculo de imposto etc.). Nestes cenários, em que o ator interage diretamente com o sistema, pode-se identificar facilmente as informações que são trocadas entre o ator e o sistema. Podemos ter uma noção clara das informações que o ator informa ao sistema, e que devem ser persistidas no computador, e as informações que o sistema deve fornecer ao ator como resposta. Pode-se ainda identificar relacionamentos entre as informações e outros sistemas ou, em outras palavras, outros bancos de dados e tabelas, com os quais provavelmente o banco de dados precisará se relacionar. Outro item muito importante a ser observado nos diagramas de casos de uso são os objetos relativos ao negócio, ou seja, os objetos da interação entre os atores. Eles são mencionados nos fluxos como sendo o objeto da interação. Como exemplo, o cliente pode selecionar o produto almejado e adicionar ao carrinho de compras.

### Diagrama de máquina de estados

O diagrama de máquina de estados é o responsável por descrever os estados de um objeto, a mudança destes estados em resposta a eventos externos e as ações que são executadas pelo objeto quando ele se encontra em um único estado. Para fazer a descrição do comportamento de um objeto, é muito comum representar todos os estados possíveis durante o seu ciclo de vida. Quando você analisa um diagrama de estado para um objeto, pode ter certeza de que você tem um **atributo** que representa o estado daquele objeto e também um **domínio** para o atributo.

O atributo pertence a uma classe de negócio do diagrama de classes. Ocorre que não é apenas isso e quase sempre temos informações que necessitam ser registradas em cada estado pelo qual o objeto passa ao longo do seu ciclo de vida. Pode ser apenas uma data, mas também pode ser que um objeto em um determinado estado tenha que realizar várias ações e que, para cada ação, seja interessante registrar as informações para consultas posteriores. Dessa forma, esses elementos de informação serão traduzidos em atributos e/ou classes do diagrama de classes.

### Diagrama de atividades

O diagrama de atividade é o responsável por descrever os fluxos de controle de uma ou mais funções do negócio. Uma atividade representa uma tarefa em um processo de negócio do mundo real ou uma operação sobre algum objeto ou classe do sistema. Neste diagrama, todas as atividades merecem muita atenção para que se identifique os elementos de informação que são manipulados pela atividade. Quase sempre é possível identificar um novo elemento de informação ou associação com outra classe do diagrama de classes.

### Exemplo de uso da linguagem UML para projeto conceitual de banco de dados

Vamos ver um exemplo de uso da linguagem UML para projeto conceitual de banco de dados. Um ponto importante é que o banco de dados a ser implementado futuramente pelo projeto físico pode ser um banco de dados relacional, objeto relacional ou orientado a objetos. A modelagem não muda com relação a nenhuma estrutura que será utilizada na construção do banco de dados.

O cenário proposto a seguir é para um sistema de informação para apoiar uma universidade nos processos de negócio de matrícula, avaliação e solicitação de bolsa de estudos. É um cenário simples que serve apenas para ilustrar a abordagem com a linguagem UML para projeto conceitual de banco de dados.

Imagine o cenário com os alunos se matriculando em disciplinas que são oferecidas e lecionadas por professores. O departamento de cadastro acadêmico é encarregado de manter um programa de disciplinas em um catálogo. Eles têm autoridade para acrescentar e excluir disciplinas, além de realizar mudanças no cadastro. Também definem limites de matrícula nas disciplinas. O departamento financeiro é encarregado de fazer o processamento de pedidos de bolsa de estudos realizados pelos alunos.

Suponha, então, que precisamos projetar um sistema para apoiar os processos de realização da matrícula, processamento de pedido de bolsa, solicitação de bolsa de estudo, manutenção do catálogo de disciplinas da universidade pelo departamento de registro acadêmico.

Suponha, ainda, que um dos requisitos do sistema seja permitir que os professores cadastrem as notas para as disciplinas que lecionam e que os alunos possam se matricular nelas e solicitar ajuda de bolsa de estudos. Ainda, suponha que é permitida a matrícula de até 50 alunos em uma disciplina, que é uma regra de negócio seguida pela universidade.

Todos os requisitos mencionados estão representados por diagramas UML que retratam diferentes dimensões que o sistema deve atender para satisfazer todos os requisitos.

O **diagrama de casos de uso** é o que permite que os atores realizem ações no sistema.

Figura 10 – Diagrama de caso de uso

A imagem conta com três bonecos que expressam os usuários do sistema e as ações que estão ocorrendo com o banco de dados. As ações são expressas por elipsóides informando as ações dos atores ao banco de dados.

Com o exemplo da figura anterior, podemos entender que ações e interações serão necessárias no banco de dados. Note que temos três atores que estão agindo com o sistema, logo estão realizando tarefas e estas tarefas estão sempre ligadas ao banco de dados.

A imagem mostra que o ator professor informa a grade e valida usuários na aplicação e estas ações são consultadas e persistidas no banco de dados, então pode-se assumir que temos as tabelas **Grade** e **Usuário** no banco de dados.

O ator aluno pode fazer a matrícula no curso e também pode pedir auxílio. Observe que o seu *log in* também é consultado no banco de dados para verificar e registrar as suas interações. Também pode-se aferir que temos mais duas tabelas no banco de dados e estas devem ser identificadas como **Matrícula** e **Auxílio**.

O último ator a interagir com o sistema é a diretora de auxílio financeiro, que pode realizar a consulta do pedido de auxílio e validar ou não este pedido.

Neste exemplo, você pode observar que vários atores podem estar interagindo e trabalhando com as tabelas no banco de dados. A leitura é feita por todos os atores em qualquer momento, mas apenas uma escrita no banco de dados é realizada por vez, para assim garantir que não vai haver sobreposição de dados na tabela ou, ainda, consulta de dados inexistentes.

(#topo)

## Modelo relacional

O modelo relacional é o modelo de dados que é representativo, ou seja, de implementação e é adequado a ser o modelo subjacente de um SGBD, que por sua vez se baseia no princípio de que completamente todos os dados estão armazenados (persistidos) em tabelas no banco de dados. A sua definição é teórica e se baseia principalmente na lógica de predicados e na teoria dos conjuntos.

## Teoria dos conjuntos

A teoria dos conjuntos é o ramo da matemática que estuda os conjuntos, que, por sua vez, tratam do comportamento de uma coleção de elementos.

No que se refere a bancos de dados, essa teoria é aplicada para extrair informações. As principais operações, chamadas de operações de conjuntos, aplicadas para a extração de dados são à união, diferença e intersecção.

### União

Esta operação permite que todos os elementos dos conjuntos sejam unidos, ou seja, agrupados. No caso de um banco de dados, todos os dados das tabelas participantes da operação seriam agrupados. Vejamos um exemplo:

Tabela de produto 1

Produto	Peso
Melão	800 g
Morango	150 g
Maça	120 g
Limão	200 g

Tabela de produto 2

Produto	Peso
Melão	800 g
Morango	150 g
Pinhão	100 g
Caqui	350 g

Agora, vamos aplicar a união entre as tabelas. Observe o resultado:

Tabela de produto 3



Produto	Peso
Melão	800 g
Morango	150 g
Maça	120 g
Limão	200 g
Pinhão	100 g
Caqui	350 g

Figura 11 – Representação do conjunto união

Fonte: adaptado de Takahashi (2009).

Dois círculos que se sobrepõem e deixam as frutas que são comuns nas duas tabelas entre eles, as demais frutas estão em círculos e não são unidas.

Esta definição de união pode ser utilizada no banco de dados e podemos perfeitamente fazer a união de tabelas utilizando comando SQL.

Na imagem anterior, temos a construção de três tabelas, e é possível unir as tabelas **professores** e **alunos**, pois elas têm a mesma quantidade de atributos. A tabela **contatos** não tem a quantidade de atributos que as demais tabelas e por isso não poderá compor a união entre as tabelas.

Figura 12 – Tabelas do banco de dados

Representação de um banco de dados composto por três tabelas (professores, alunos e contatos).

Agora, vamos utilizar o comando SQL para unir as duas tabelas do banco de dados.

Ao abrir a aba **SQL**, digite o comando:

```
Select * from alunos union select * from professores;
```

É fácil perceber que temos a palavra **union** entre os comandos de **select** do banco de dados e que logo será feita a união entre as tabelas. Observe o resultado do comando SQL na figura a seguir.

Figura 13 – Representação da união de tabelas

Representação do comando SQL de “select” e a saída de todos os registros inseridos nas tabelas professores e alunos.

### Diferença

Esta operação permite que todos os elementos diferentes dos conjuntos sejam selecionados. Uma operação de diferença pode extrair todos os dados da primeira tabela que não estejam incluídos na segunda. Nas figuras, observe a aplicação da operação da diferença entre os dados dos conjuntos e também entre as tabelas. Vejamos um exemplo:

Tabela de produto 1

Produto	Peso
Melão	800 g
Morango	150 g
Maça	120 g
Limão	200 g



Tabela de produto 2

Produto	Peso
Melão	800 g
Morango	150 g
Pinhão	100 g
Caqui	350 g

Agora vamos aplicar a diferença entre as tabelas, observe o resultado:

Tabela de produto 1

Produto	Peso
Maça	120 g
Limão	200 g

Tabela de produto 2

Produto	Peso
Pinhão	100 g
Caqui	350 g

Figuras 14 e 15 – Representação da aplicação da diferença entre as tabelas de produtos 1 e 2

Fonte: adaptado de Takahashi, 2009.

a primeira figura contém dois círculos sobrepostos e apenas o círculo à direita contém todos os produtos da tabela 2, enquanto que o círculo à esquerda contém apenas os produtos da diferença (pinhão e caqui). A segunda figura é o complemento da figura anterior, ou seja, agora o círculo da tabela 1 é que contém todos os produtos expressos na tabela 1.

O banco de dados MySQL não trata as operações de diferença diretamente no SQL, mas podemos utilizar o comando **JOIN** (junção) para efetuar a diferença entre duas tabelas. Para este exemplo, foi adicionado mais um nome nas tabelas **professores** e **alunos** e este nome foi o mesmo nas duas tabelas, assim será possível visualizar a diferença entre os dois conjuntos. Observe as figuras a seguir:

Figuras 16 – Seleção dos dados da tabela professores

A imagem é um *grid* com o resultado da consulta SQL, “Select \* from professores”. A saída contém três linhas com os dados inseridos no banco de dados.

Figuras 17 – Seleção dos dados da tabela alunos

A imagem é um *grid* com o resultado da consulta SQL, “Select \* from alunos”. A saída contém três linhas com os dados inseridos no banco de dados.

Com a inserção do nome **Maria** nas duas tabelas, podemos então conferir a saída do comando **left join** entre as tabelas:

Figuras 18 – Seleção dos dados das tabelas aluno e professores

A imagem é um *grid* com o resultado da consulta SQL, “select a.nome as NomeAluno, b.nome as NomeProfessor FROM alunos a left JOIN professores b ON a.nome = b.nome and b.nome = a.nome”. A saída contém três linhas com os dados inseridos no banco de dados.

Com o comando **left join**, pode-se então simular a diferença entre as tabelas.

### Intersecção

Esta operação permite que todos os elementos iguais dos conjuntos sejam selecionados. Uma operação de intersecção pode extrair todos os dados da primeira tabela que estejam incluídos na segunda. Observe na figura a aplicação da operação de intersecção entre os de dados dos conjuntos e também entre as tabelas.

Tabela de produto 1

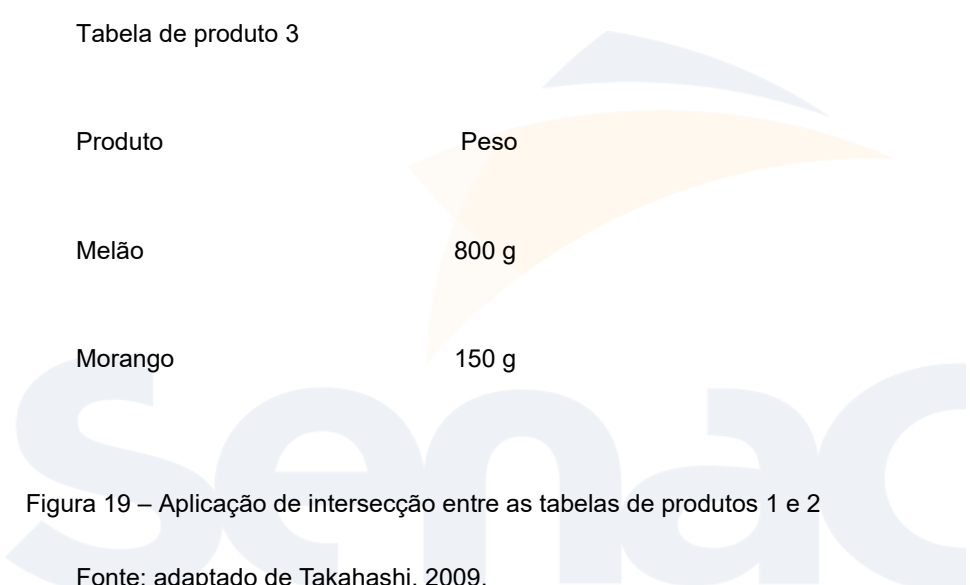
Produto	Peso
Melão	800 g
Morango	150 g
Maça	120 g
Limão	200 g

Tabela de produto 2

Produto	Peso
Melão	800 g
Morango	150 g
Pinhão	100 g
Caqui	350 g

Agora vamos aplicar a intersecção entre as tabelas, observe o resultado:

Tabela de produto 3



Produto	Peso
Melão	800 g
Morango	150 g

Figura 19 – Aplicação de intersecção entre as tabelas de produtos 1 e 2

Fonte: adaptado de Takahashi, 2009.

Dois círculos sobrepostos que representam as tabelas e agora o destaque está apenas para os produtos que são iguais nas duas tabelas (melão e morango).

Utilizando ainda as figuras anteriores, podemos fazer a intersecção entre as tabelas do banco de dados. Observe a figura a seguir:

Figura 20 – Aplicação de intersecção entre as tabelas

A imagem é um *grid* com o resultado da consulta SQL, “select a.nome as NomeAluno, b.nome as NomeProfessor FROM alunos a JOIN professores b ON a.nome = b.nome and b.nome = a.nome”. A saída contém apenas uma linha com os dados inseridos no banco de dados que são iguais nas duas tabelas.

## Principais tipos de dados

A escolha do tipo de dado é um ponto extremamente importante no que se refere ao desenvolvimento e à construção das tabelas do banco de dados. Quando se escolhe o tipo de dado, está se determinando como a informação deve ser inserida e quanto espaço ela poderá ocupar. Essa preocupação, que inicialmente pode parecer irrelevante, é uma iniciativa de prevenção de falhas do banco de dados.

### Dados numéricos

- ◆ **Smallint:** é um número inteiro de 16 *bits* que permite representar até 65.535 números, entre positivos e negativos.
- ◆ **Integer** ou **int:** é um número inteiro de 32 *bits* que permite representar até 4.294.967.295 números, entre positivos e negativos.
- ◆ **Bigint:** é um número inteiro de 64 *bits* que permite representar até 18.446.744.073.709.551.615 números, entre positivos e negativos.
- ◆ **Real:** é um número fracionário de precisão simples, ou seja, ocupa o tamanho máximo de 32 *bits*.
- ◆ **Decimal:** é um número fracionário com casas decimais exatas, ou seja, pode-se controlar exatamente a quantidade de casa inteiras e decimais de um número. A declaração decimal(10,2), por exemplo, indica que temos dez dígitos de representação numérica, porém, dois deles servirão para a parte fracionária do número.

### Dados alfanuméricos

- ◆ **Char:** faz alocação fixa da quantidade de caracteres a serem armazenados. Por exemplo, se declaramos um char(200) e, dentro desse campo, escrevermos a palavra “João”, ela ocupará quatro posições de caracteres para ser armazenada. Entretanto, como alocamos 200 posições, será reservado em disco espaço para 200 caracteres, independentemente da quantidade deles que for ocupada.
- ◆ **Varchar:** faz alocação variável da quantidade de caracteres a ser armazenada. Se declaramos varchar(200), por exemplo, e dentro dele escrevermos a palavra “João”, das 200 posições reservadas para armazenamento, apenas quatro serão ocupadas. As outras 196 posições que estiverem sem conteúdo serão compactadas, reduzindo-se assim o espaço de armazenamento em disco.

### Dados de tempo

- ◆ **Date:** é utilizado para armazenar datas (padrão americano: AAAA-MM-DD). Exemplo: 2008-01-11.
- ◆ **Time:** é utilizado para armazenar horas (padrão americano: HH:MM:SS). Exemplo: 22:54:32.

## Normalização do banco de dados

O processo de normalização deve sempre fazer parte do projeto de banco de dados. Trata-se da organização entre as tabelas, o que garante um gerenciamento simples, eficiente e mais seguro.

As fases normais devem ser aplicadas ao diagrama para que se possam procurar problemas na estruturação e modelagem.

O foco da normalização são as características de cada entidade. Ela representa uma microvisualização das entidades do diagrama.

O processo de normalização pode resultar em entidades e atributos adicionais. Para ilustrar o papel adequado da normalização no processo do projeto, é necessário reavaliar as regras de negócios da empresa.

O processo de normalização divide-se em cinco fases, chamadas de formas normais: 1FN, 2FN, 3FN, 4FN, 5FN.

## Integridade referencial

A integridade referencial está ligada a toda garantia de que um valor que aparece em uma relação para determinado conjunto de atributos também apareça para certo conjunto de atributos em outra relação.

As regras de chaves mantêm a integridade referencial das tabelas por meio da ligação entre as colunas que as receberam e a validação de valores que poderão ser inseridos.

Observe o comportamento dos campos que são compostos por chaves.

### Chave primária (*primary key*)

Uma chave primária é uma regra implementada em uma coluna ou em um conjunto de colunas de forma a garantir que os valores contidos nelas sejam únicos, ou seja, que esses valores nunca se repitam.

### Chave alternativa ou candidata (*unique key*)

Algumas colunas, que naturalmente apresentam característica de informação única, como CPF ou CNPJ, podem ficar de fora da regra de chave primária. Para garantir que os valores inseridos nessas colunas sejam únicos, podemos implementar a regra de chave alternativa, que também garante a unicidade das informações na coluna (ou colunas) que recebem esta regra.

### Chave estrangeira (*foreign key*)

Uma chave estrangeira é uma regra que pode determinar o comportamento de uma ou mais colunas fazendo com que ela(s) referencie(m) as informações existentes em uma chave primária. Isto é, toda coluna que recebe regra de chave estrangeira deve ter recebido a regra de chave primária em sua tabela de origem, pois isso permite estabelecer relacionamentos entre tabelas do banco de dados.

Departamento

Código	Nome
4620	Fundamentos da computação
4622	Computação aplicada

## Empregado

Id	Nome	Código	Superior
1	Ir. Clotet		
2	Avelino	4620	1
3	Rodrigo	4622	2

A tabela **departamento** tem como chave primária (PK) o campo **código**. Na tabela **empregado**, o campo **ID** recebe regra de chave primária (PK). O campo **código** da tabela **empregado**, recebe chave estrangeira (FK). Há uma ligação entre o campo **código** da tabela **departamento** e o campo **código** da tabela **empregado**, a fim de apresentar a integridade referencial entre os campos de ambas as tabelas.

(#topo)

## Componentes de sistemas de bancos de dados

Agora que já entendemos como planejar o banco de dados utilizando técnicas e ferramentas assegurando assim a integridade, a consistência e as regras, chegou a hora de fazer o *download* e a instalação do banco de dados.

Neste processo, iremos fazer a instalação pela ferramenta XAMPP e trabalhar diretamente no navegador e também da ferramenta MySQL Workbench, que é um gerenciador do banco de dados MySQL.

### XAMPP

O XAMPP é um servidor independente (*software* livre), que consiste principalmente na base de dados MySQL o servidor *web* Apache e os interpretadores para linguagens de *script*: PHP e Perl. O nome vem da abreviação de X (para qualquer dos diferentes sistemas operativos), A do Apache, M do MySQL, P do PHP, e o último P da linguagem Perl. É um método que torna extremamente fácil para os desenvolvedores criar um servidor *web* local para fins de teste.

Por meio do XAMPP podemos acessar o servidor *web* e testar nosso banco de dados, mas primeiramente precisamos fazer a instalação do XAMPP.

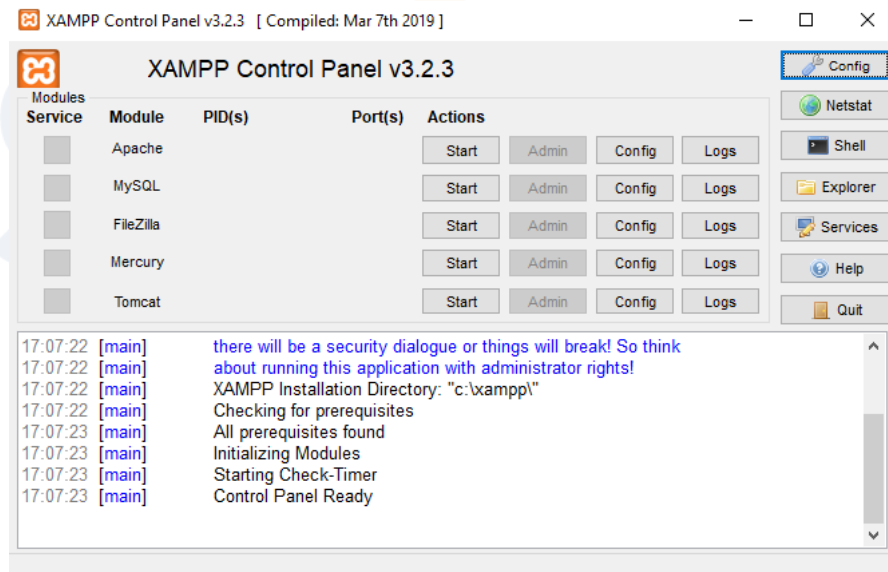
A distribuição do XAMPP é de responsabilidade da Apache, podendo ser baixado diretamente pelo *site* da distribuidora.

Basta você escolher o seu sistema operacional e clicar em baixar.

Após o *download* do executável, basta seguir os passos e fazer a instalação do XAMPP.

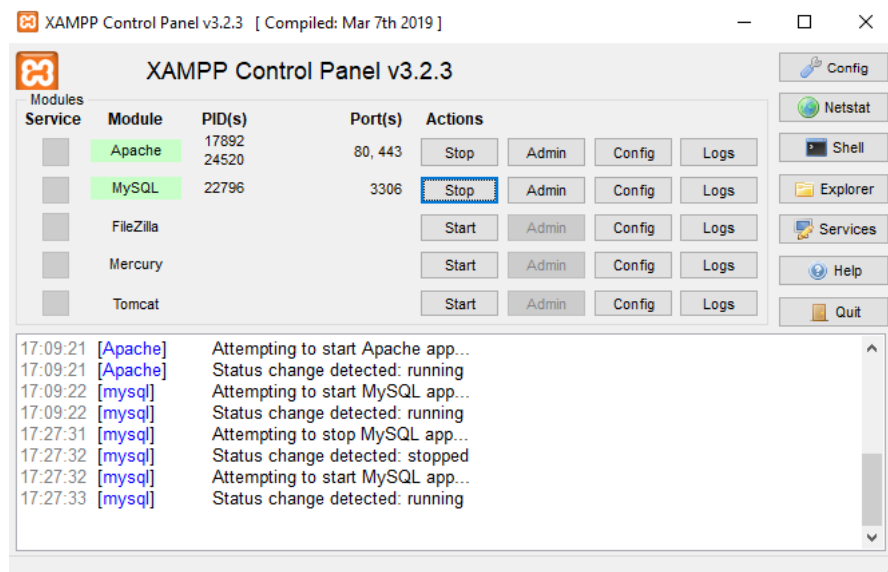
Após a instalação completa, é possível acessar o banco de dados pela tela da aplicação, mas primeiramente vamos inicializar o aplicativo.

Com o aplicativo inicializado, clique no botão **start** na linha que compreende o Apache e o MySQL.



Tela inicial do programa XAMPP, com botões para iniciar e configurar o serviço do Apache e MySQL.

O módulo Apache ficou com a cor verde e o MySQL também na cor verde, significando que o XAMPP está rodando na máquina. Agora, clique em **Admin**, na linha do MySQL.



A imagem a tela inicial do programa XAMPP. Agora o XAMPP foi iniciado e ficou marcado com a cor verde.

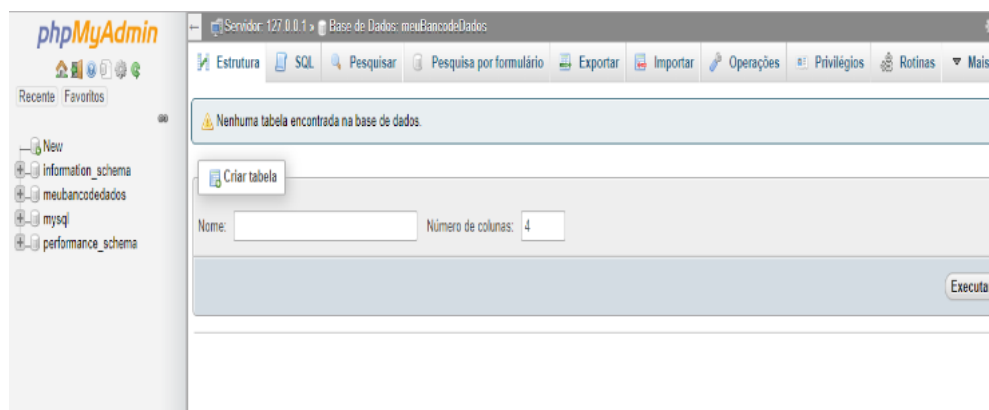


Abriremos o navegador com o gerenciador do banco de dados. Com o phpMyAdmin aberto, podemos começar a construir o nosso banco de dados. Insira o nome do banco de dados e clique em **Criar**.



Navegamos no interior da tela phpMyAdmin e aparece um *grid* com informações do banco de dados.

Para o exemplo, criaremos o banco de dados com nome **meuBancodeDados**. Após, o nome aparece no menu lateral à esquerda e logo em seguida abre uma nova tela, que está esperando receber a criação das tabelas do banco de dados.



Tela com o nome do banco de dados criado exposto na parte esquerda, no menu de navegação.

Para criar a tabela, digite o nome dela e informe o número de colunas que ela deverá ter. Chamaremos a tabela de **minhaTabela**, neste exemplo. Logo, registre os nomes dos campos e seus tipos de dados. Nesta parte, também definimos a nossa chave primária e o campo de autoincremento.

Tela composta por grids onde podemos criar nossas tabelas e nossos campos.

Note que o primeiro campo, que chamei de **idminhaTabela**, tipo **INT**, é o que contém a chave primária e é autoincremento. Abaixo, estão os campos **nome** e **endereco** (sem cedilha) do tipo **varchar** e com tamanho 80 e, por último, o campo **telefone**, com tipo de dado também **varchar**, mas com tamanho 15. Após concluir os registros, clique em **Guarda**.

Tela composta por *grids* em que podemos criar nossas tabelas e nossos campos. Nesta tela, estamos definindo a nossa chave primária e a coluna que será autoincremento.

Para visualizar o resultado da tabela, clique no menu **Estrutura**. Temos a nossa tabela corretamente criada e podemos inserir os dados para posterior consulta.

#	Nome	Tipo	Agrupamento (Collation)	Atributos	Nulo	Predefinido	Comentários	Extra	Ações
1	idminhaTabela	int(11)			Não	None		AUTO_INCREMENT	<a href="#">Muda</a> <a href="#">Elimina</a> <a href="#">Mais</a>
2	nome	varchar(80)	latin1_swedish_ci		Não	None			<a href="#">Muda</a> <a href="#">Elimina</a> <a href="#">Mais</a>
3	endereco	varchar(80)	latin1_swedish_ci		Não	None			<a href="#">Muda</a> <a href="#">Elimina</a> <a href="#">Mais</a>
4	telefone	varchar(15)	latin1_swedish_ci		Não	None			<a href="#">Muda</a> <a href="#">Elimina</a> <a href="#">Mais</a>

Índices	Ações	Nome da chave	Tipo	Único	Pacote	Coluna	Quantidade	Agrupamento (Collation)	Nulo	Comentário
PRIMARY	<a href="#">Edita</a> <a href="#">Elimina</a>	PRIMARY	BTREE	Sim	Não	idminhaTabela	0	A	Não	

Tela onde mostra a estrutura do banco de dados criado. Estão listados todos os campos criados e seus respectivos tipos de dados.

## MySQL workbench

### ◆ **Download e instalação:**

Agora, vamos fazer o *download* da ferramenta do MySQL Workbench que se encontra disponível no *site* do MySQL Workbench.

Selecione o seu sistema operacional e clique em **Download**. Após, preencha o cadastro da empresa Oracle, que é obrigatório para seguir com o *download*. Após baixar o executável, realize a instalação, clicando em **Next**.

**MySQL Workbench 8.0.17**

Select Operating System:  
Microsoft Windows

[Looking for previous GA versions?](#)

**Recommended Download:**

**MySQL Installer for Windows**  
All MySQL Products. For All Windows Platforms. In One Package.  
Starting with MySQL 5.6 the MySQL Installer package replaces the standalone MSI packages.

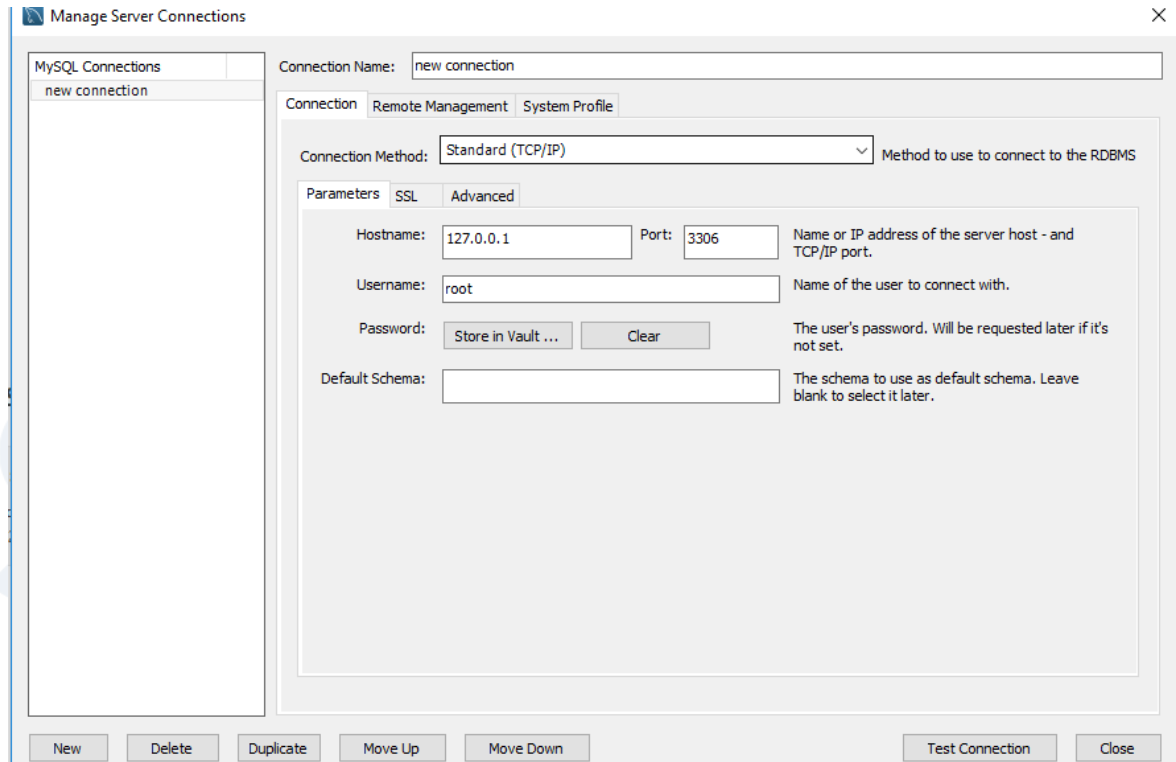
**Windows (x86, 32 & 64-bit), MySQL Installer MSI** [Go to Download Page >](#)

**Other Downloads:**

Windows (x86, 64-bit), MSI Installer	8.0.17	35.1M	<a href="#">Download</a>
--------------------------------------	--------	-------	--------------------------

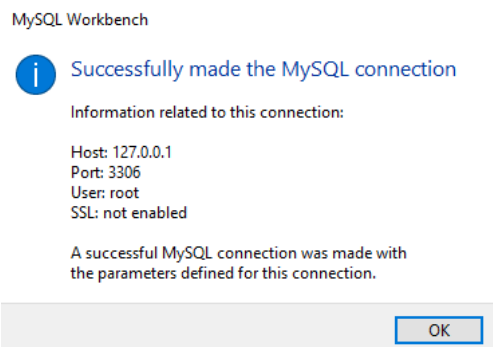
Tela do navegador onde escolhemos o sistema operacional para download da ferramenta MySQL Workbench.

Terminada a instalação, você precisará instanciar uma conexão. Adicione um nome à sua conexão e clique em **Test Connection**.



Tela em que aparecem diversas caixas de texto no qual devemos configurar a conexão com o banco de dados.

Se a configuração realizada estiver correta, aparecerá a tela a seguir. Clique em **Close**.



Tela em que é apresentado o resultado da conexão.

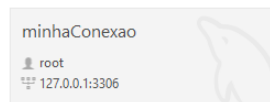
Agora, é só clicar na conexão que você acabou de configurar.

# Welcome to MySQL Workbench

MySQL Workbench is the official graphical user interface (GUI) tool for MySQL. It allows you to design, create and browse your database schemas, work with database objects and insert data as well as design and run SQL queries to work with stored data. You can also migrate schemas and data from other database vendors to your MySQL database.

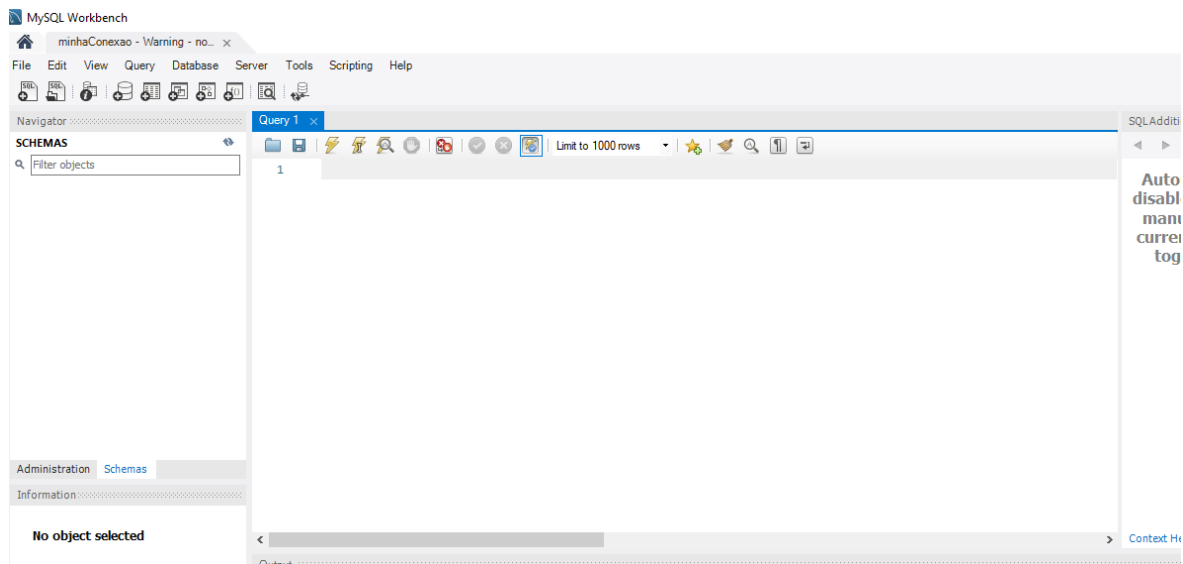
[Browse Documentation >](#)[Read the Blog >](#)[Discuss on the Forums >](#)

## MySQL Connections



Tela em que aparece a configuração da conexão criada para efetuar a entrada no banco de dados.

Então, abrirá a tela do MySQL Workbench:

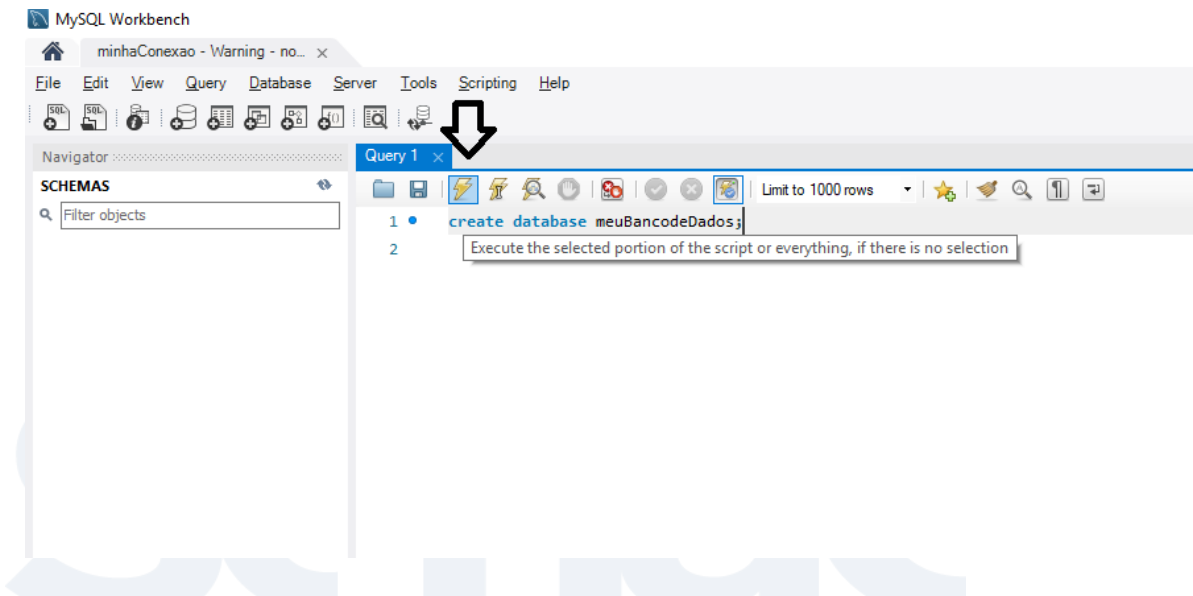


Tela inicial do MySQL Workbench na qual temos um menu lateral esquerdo que se pode navegar entre as funcionalidades e o banco dados criados e uma caixa de texto grande ao centro, em que digitamos os comandos SQL.

### ◆ Criação do banco de dados

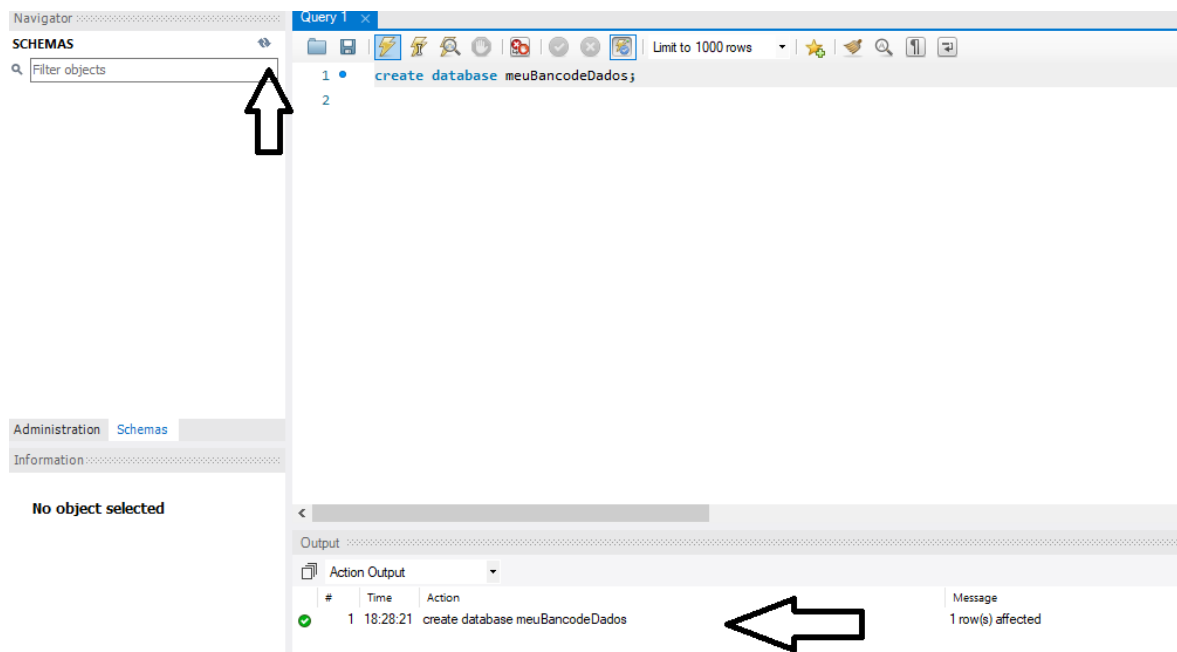
Para fazer a criação do banco de dados no MySQL Workbench, precisamos digitar os *scripts* de criação do banco de dados e das tabelas, conforme você já aprendeu na UC **Codificar aplicações web**.

Para lembrar, vamos utilizar o comando **create** para criação.



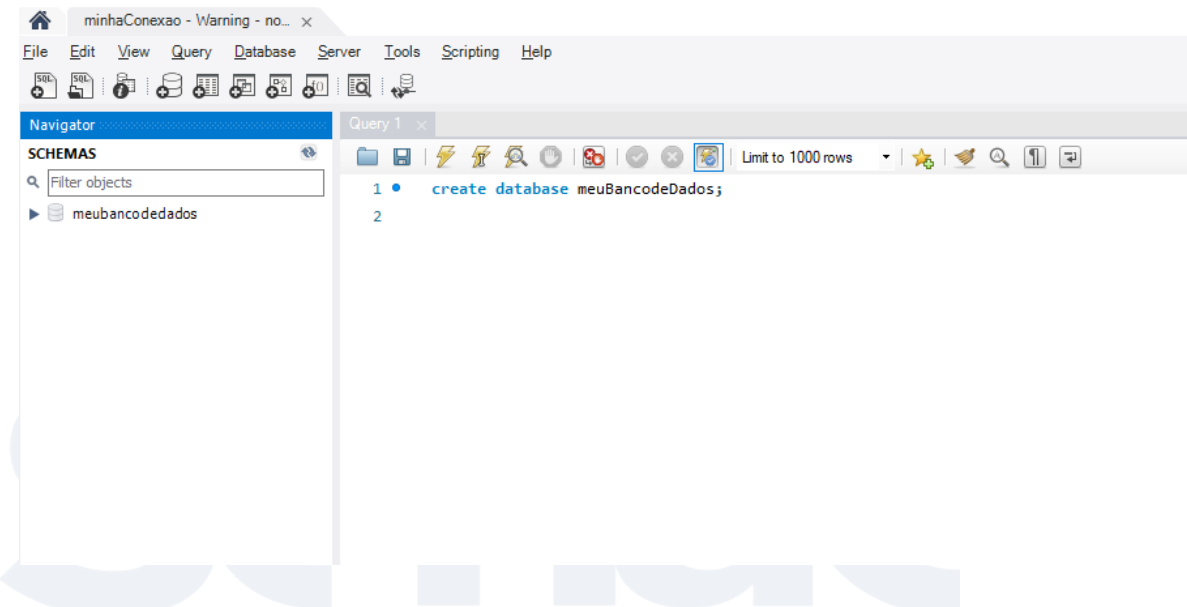
Tela em que está sendo digitado o comando SQL para criação do banco de dados na caixa de texto central.

Após a inserção do *script* de criação do banco de dados, **create database meuBancodeDados**, clique no ícone para executar o *script*. Se tudo ocorrer de forma satisfatória, o banco de dados será criado.



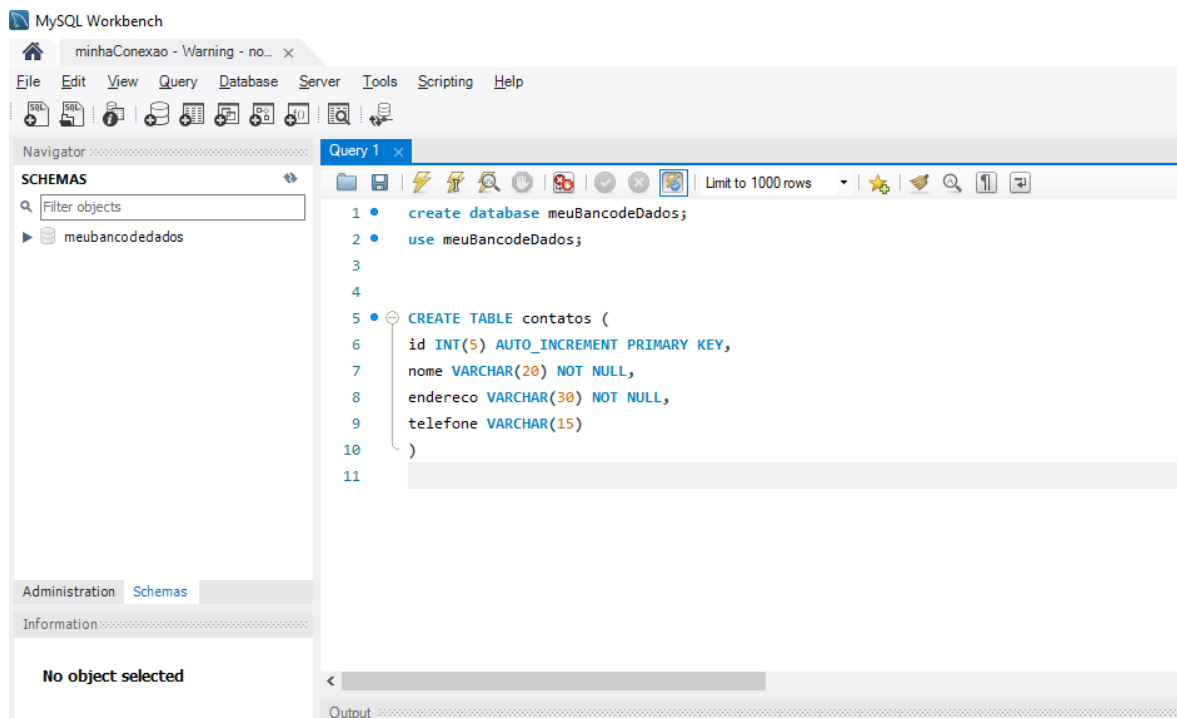
Tela em que aparece um *grid* na parte inferior com a ação de criação do banco de dados resultando em êxito.

Agora, clique no atualizar e o banco de dados irá aparecer na tela.



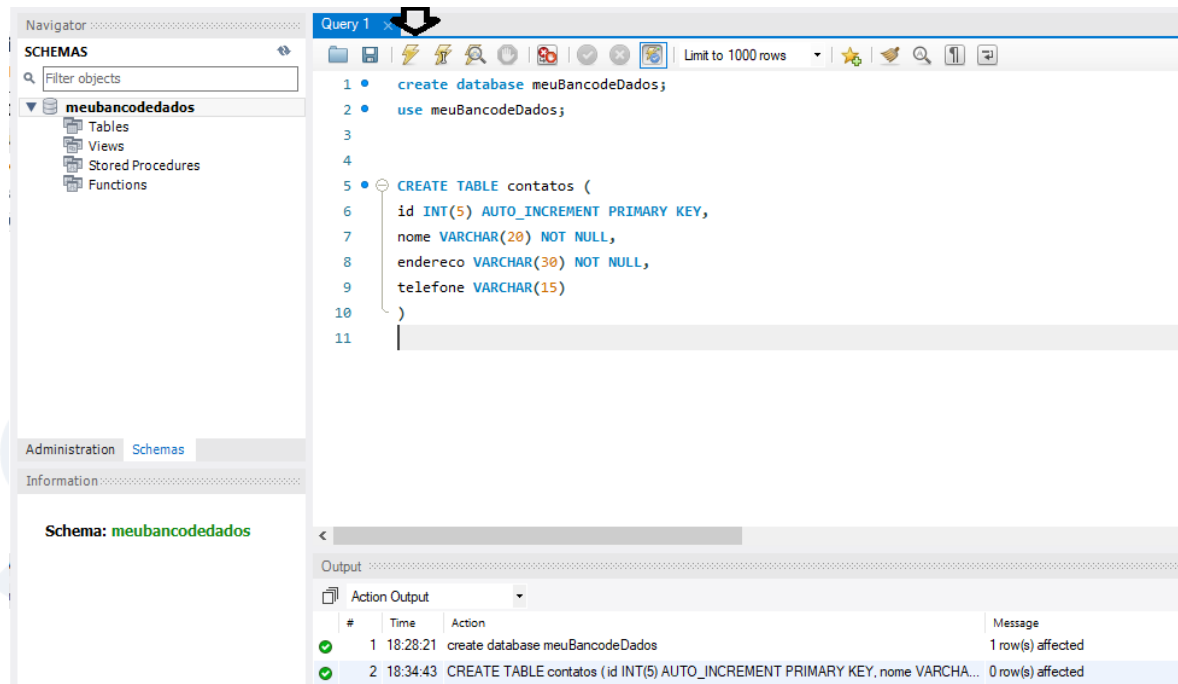
Tela em que aparece no lado esquerdo o nome do nosso banco de dados recentemente criado.

O banco de dados está aparecendo no menu lateral esquerdo, então podemos criar as nossas tabelas. Lembre-se de que novamente iremos precisar de *script* para criação das tabelas.



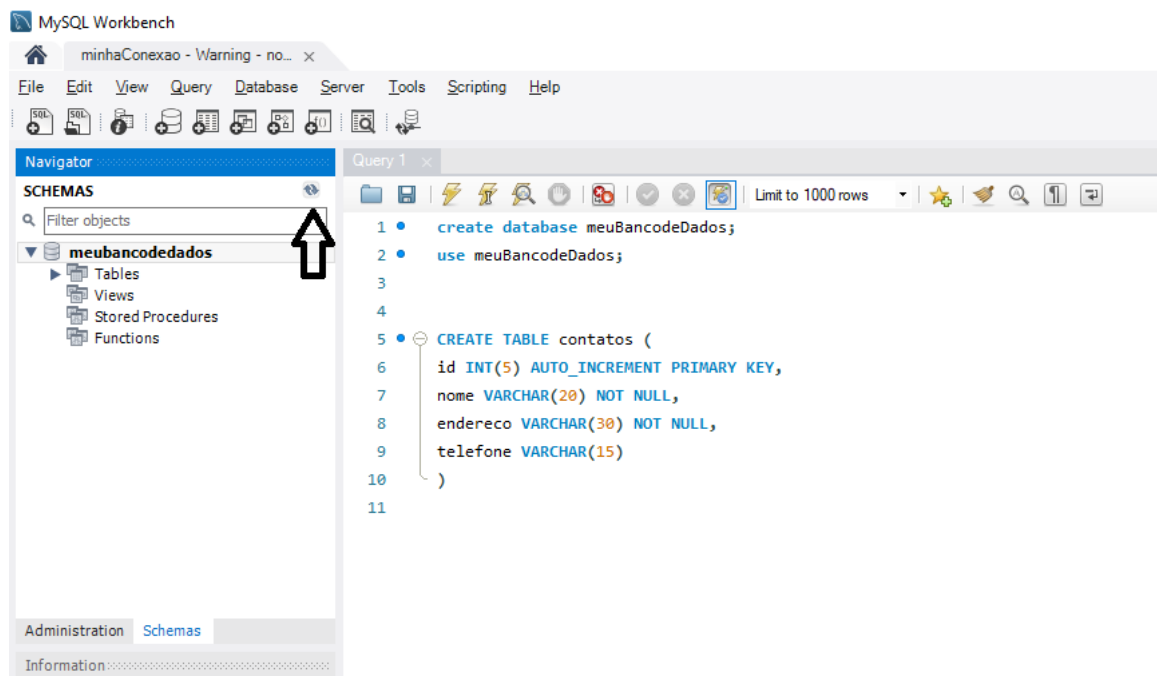
Na tela central, que contém a caixa de texto, é digitado o comando SQL para criação da tabela do banco de dados.

Após a inserção do *script*, novamente clicamos no botão **executar** para rodá-lo.



Tela em que aparece um *grid* na parte inferior com a ação de criação da tabela do banco de dados resultando em êxito.

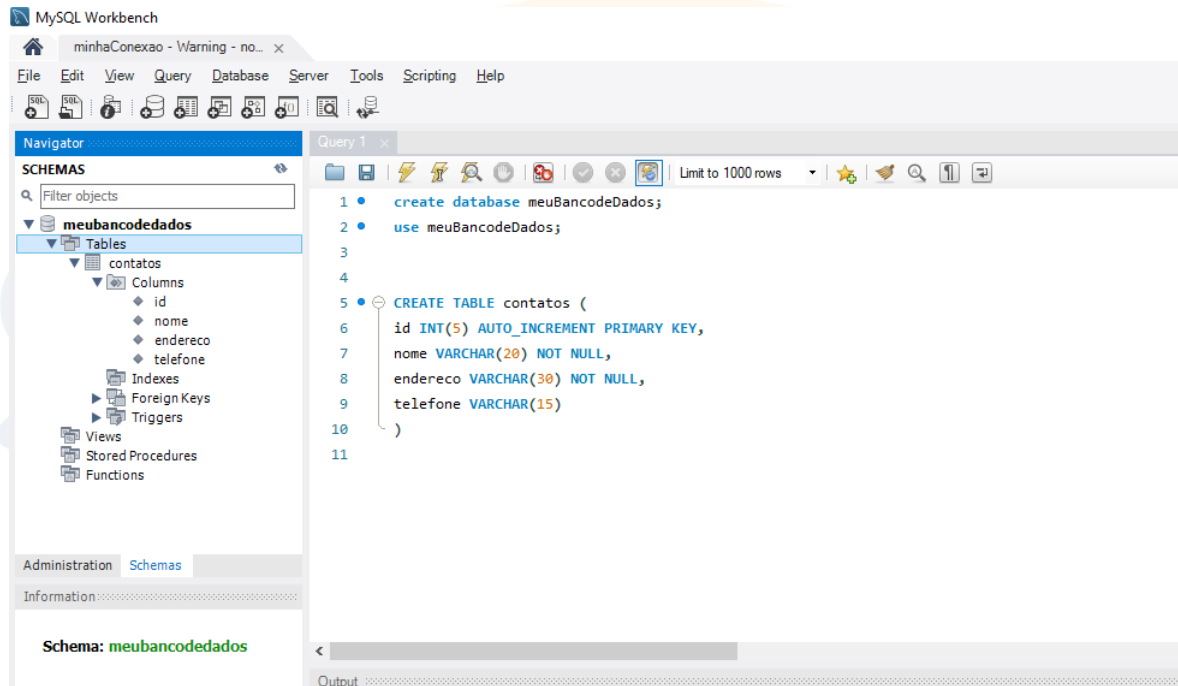
Observe que foi a tabela foi criada, conforme mensagem do console de saída do MySQL Workbench. Ainda precisamos atualizar o banco de dados para visualizar a tabela.



Após ser apresentado o *grid* de criação da tabela do banco de dados com êxito, clicamos no botão “refresh” para atualizar o banco de dados. Na lateral da imagem, aparece o banco de dados criado e incorporado a ele a tabela recém construída.



Agora que a tabela está criada, apareceu uma seta no menu **Tables** indicando que há alguma tabela criada na raiz do menu. Então, podemos explorar e verificar que a tabela, ou tabelas, sempre ficam dentro do menu **Table**.



Agora, clicamos no banco de dados e na tabela e podemos observar os campos que foram inseridos na tabela.

Neste material, você aprendeu sobre as principais rotinas e cuidados que devemos ter na criação do banco de dados.

Você estudou as abordagens do modelo conceitual e relacional de banco de dados e, por fim, aprendeu a fazer a instalação e a configuração do banco de dados MySQL com o gerenciador do XAMPP e do MySQL WorkBench.

(#topo)