

# Conceitos de Banco de Dados

## Aula 02 - Introdução à Linguagem SQL e Sistemas Gerenciadores de Banco de Dados

# Apresentação

---

Nesta aula, você vai aprender o que é SQL, por que ela foi criada e para que serve. Vai estudar algumas informações sobre seu histórico e suas características. Estudará também sobre Sistemas Gerenciadores de Banco de Dados (SGBD), que serão os softwares que você vai utilizar para criar e manipular seus bancos de dados. E, para finalizar, você vai aprender a instalar e a configurar o MySQL, um SGBD muito utilizado em aplicações para web.

Além disso, você terá a oportunidade de conhecer alguns dos conceitos importantes relacionado aos bancos de dados, as relações e o significado do uso de chaves primárias e de chaves estrangeiras.



**Vídeo 01** - Apresentação

## Objetivos

- Entender o que é SQL e para que é usado.
- Entender os conceitos de relações/tabelas e tuplas.
- Entender os conceitos de chaves primárias e chaves estrangeiras.
- Conhecer os principais Sistemas Gerenciadores de Banco de Dados.
- Identificar as principais diferenças dos Sistemas Gerenciadores de Banco de Dados.
- Instalar e configurar o MySQL.

# 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 precursores que implementaram esse modelo foi o System R (IBM) – cuja sua história você estudou 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 (que você irá conhecer mais adiante) usa o nome “tabela” em vez 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", em que seriam armazenadas informações sobre os diversos empregados. No entanto, como posso armazenar as informações em uma tabela? Para responder tal pergunta, vamos conhecer o conceito de atributo.

- **Atributos**

Atributos são todas as informações que existem em uma tabela. Essas informações são chamadas informalmente de campos. Exemplos: 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 1 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 em um site de busca, recebemos como resposta as tuplas do banco de dados daquele site, que estão relacionadas de alguma forma com o texto procurado. Informalmente, as tuplas são também chamadas pelos desenvolvedores de registros.

**Figura 01** - Os atributos e as tuplas de uma tabela Empregado.

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

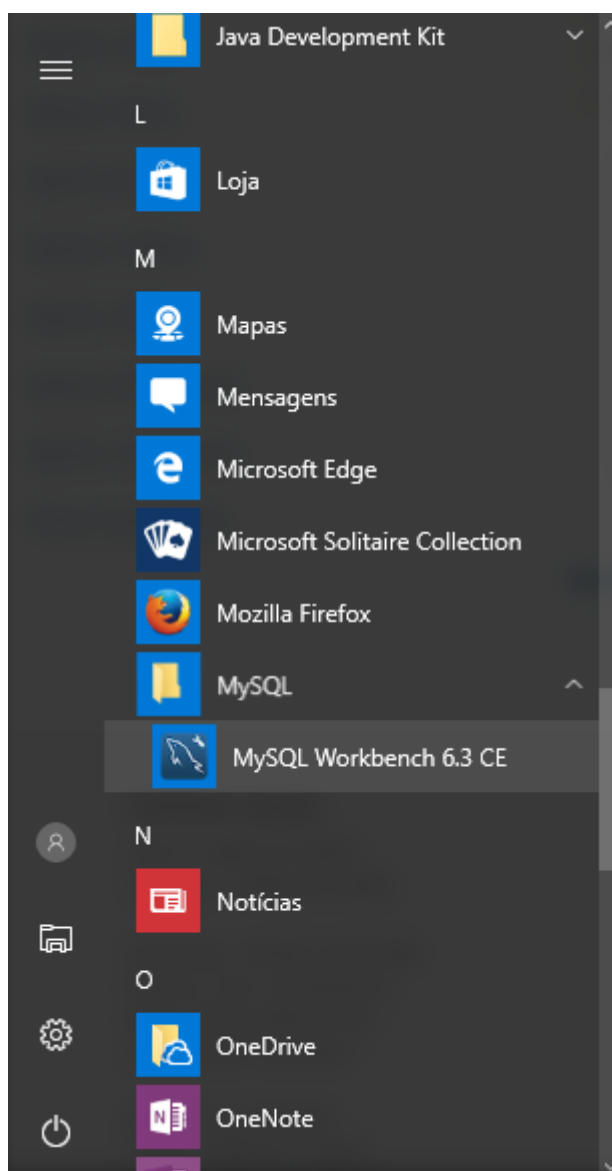
Depois de aprender o significado dos conceitos Tabela, Atributo, Domínio e Tuplas, é importante agora aprender como utilizar a ferramenta MySQL Workbench para definir tais conceitos no Modelo Relacional.

## Utilizando a Ferramenta *Mysql Workbench*

---

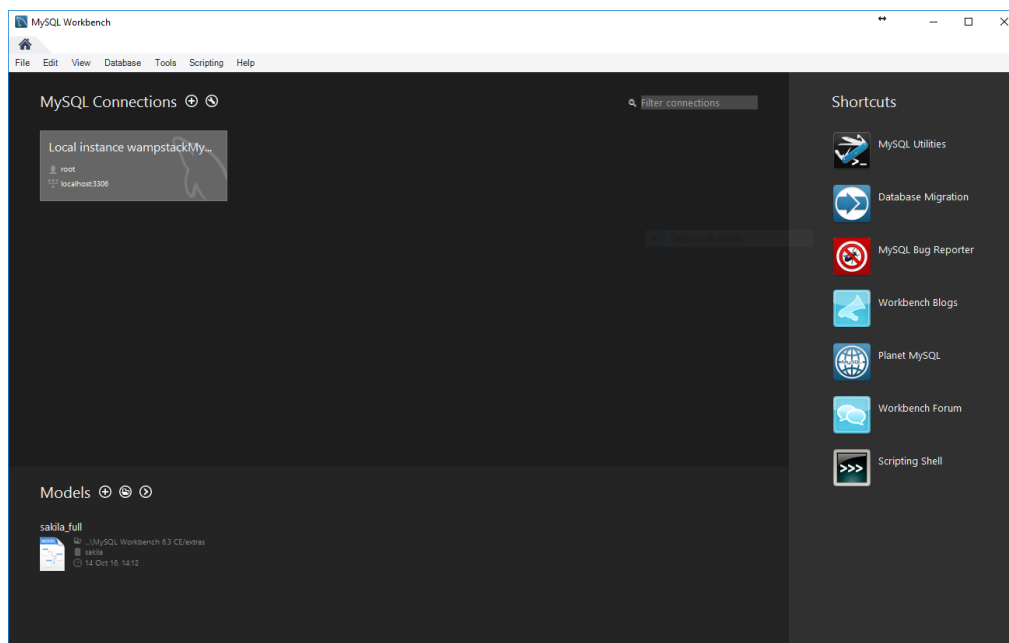
Antes de tudo, vamos abrir a ferramenta *Mysql Workbench* que encontra-se no menu iniciar do Windows, conforme a **Figura 2**. Caso você use Linux, a ferramenta provavelmente será instalada no menu de aplicações.

**Figura 02** - Menu Iniciar *MySQL Workbench* no Windows.

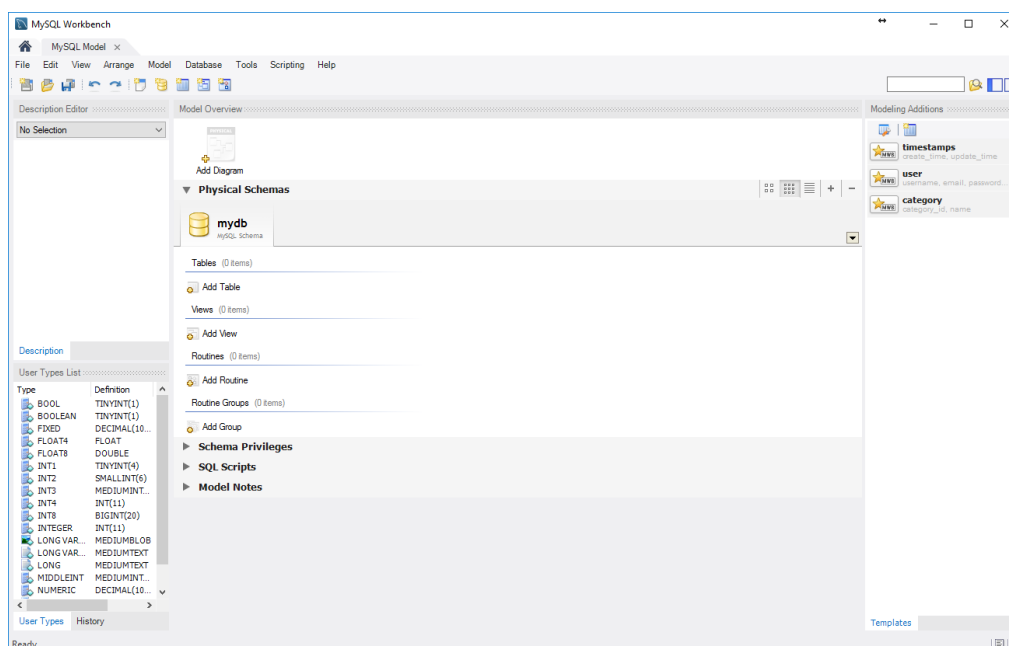


Na **Figura 3**, temos a tela inicial do *MySQL Workbench*. Para começar, vamos escolher a opção *Create New Err Model*. Escolhendo essa opção, avançamos para uma próxima tela, na qual temos a opção *Add Diagram*. Essa tela pode ser visualizada pela **Figura 4**. Clicando duas vezes em *Add Diagram*, abrimos nossa área para criarmos o Modelo Relacional.

**Figura 03** - Tela de abertura do *MySQL Workbench*.



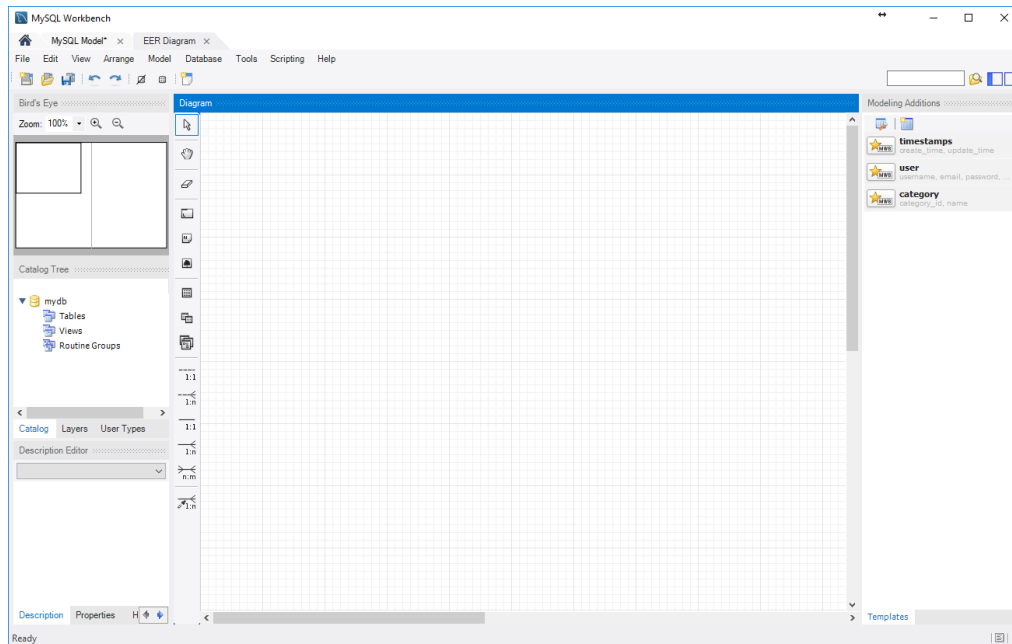
**Figura 04** - Ambiente para criação dos diagramas no *MySQL Workbench*.



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 5**, temos o ambiente da ferramenta em que iremos criar nosso Modelo Relacional. O modelo é criado na parte branca, que ocupa a maior parte do lado inferior direito da ferramenta.



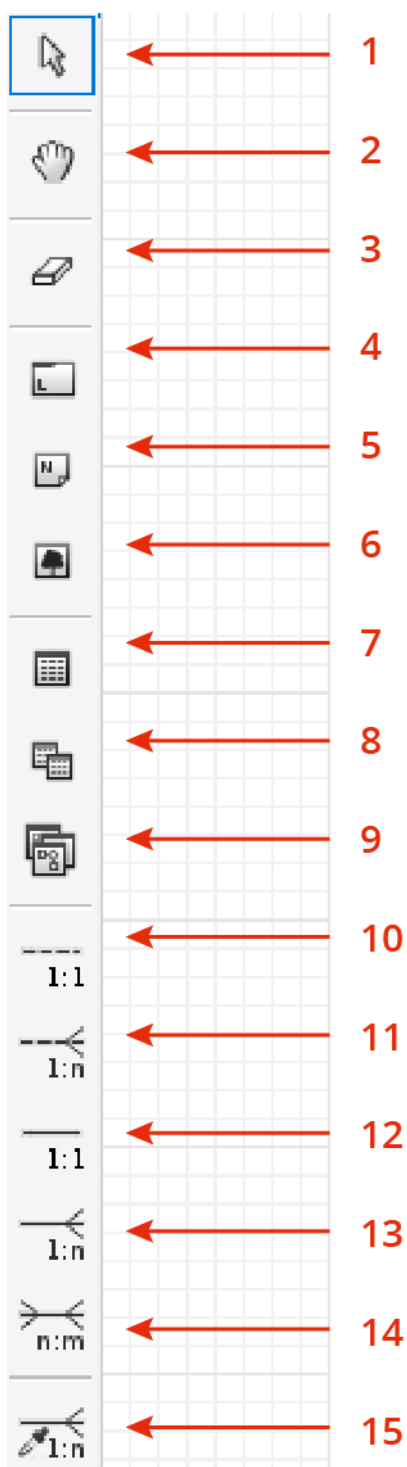
**Figura 05** - Ambiente de trabalho para desenvolver o Modelo Relacional no *MySQL Workbench*.



E agora, como criamos uma tabela? Vejamos o passo a passo a seguir:

1. Na barra mostrada pela **Figura 6**, 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 à direita da barra.

**Figura 06** - Barra de Ferramentas do *MySQL Workbench*.



2. Ao clicar com o botão direito do mouse em cima da tabela criada, você verá a opção **Edit 'Table 1'**. Quando clicar nessa opção, vai aparecer, na parte inferior da tela, uma aba em que existe a opção Table. Nessa aba, você irá ver um campo de texto com nome **Table Name**, no qual

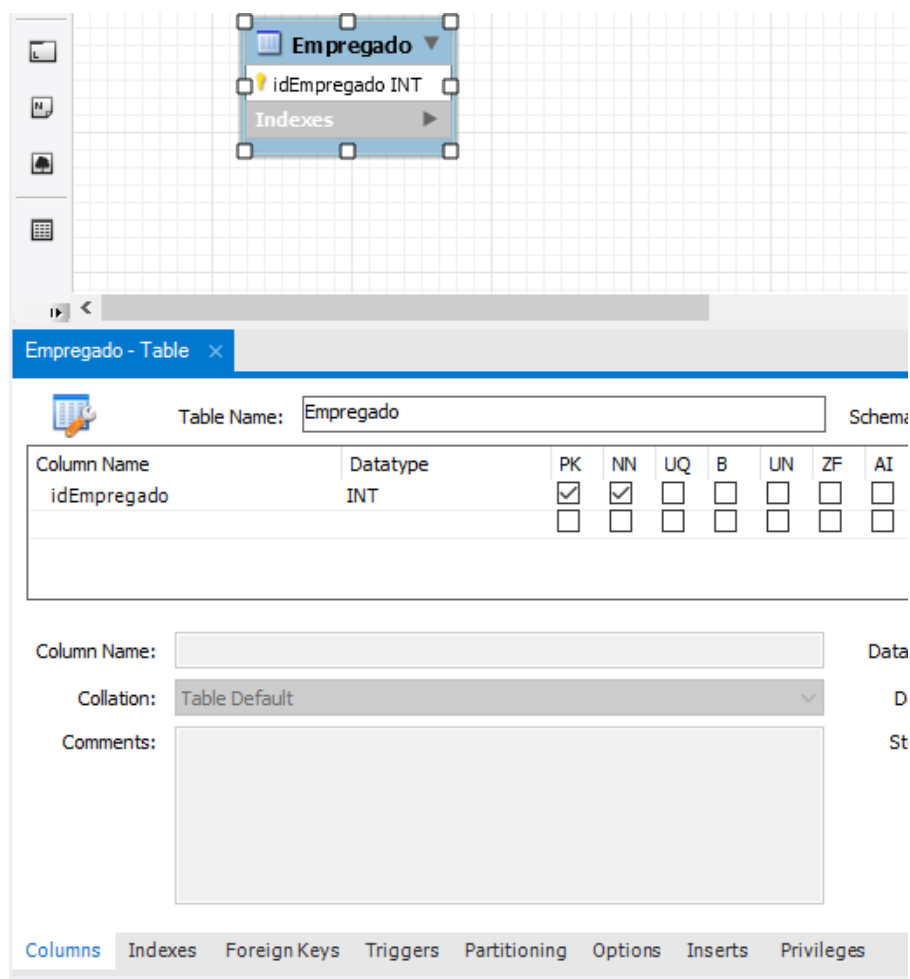
colocamos o nome da tabela. Note que Table em inglês equivale à tabela em português. Para o nosso exemplo, você deve fornecer o nome **Empregado**. Ao final dessa operação, você deve obter algo parecido com a **Figura 7**.

**Figura 07** - Exemplo de criação da Tabela no *MySQL Workbench*.



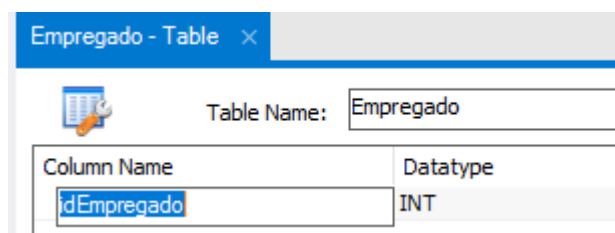
3. Agora, devemos definir os atributos da tabela. Para isso, devemos clicar na aba inferior chamada *Columns*. Você deve informar o nome do atributo e o tipo em cada linha da tabela. Note que ao clicar na aba *Columns*, o *MySQL Workbench* cria automaticamente um atributo com nome `idEmpregado` com tipo `INT` (**Figura 8**).

**Figura 08** - 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 em que está o nome do atributo idEmpregado. Quando o cursor ficar como o mostrado na **Figura 9**, você pode digitar o nome do atributo desejado.

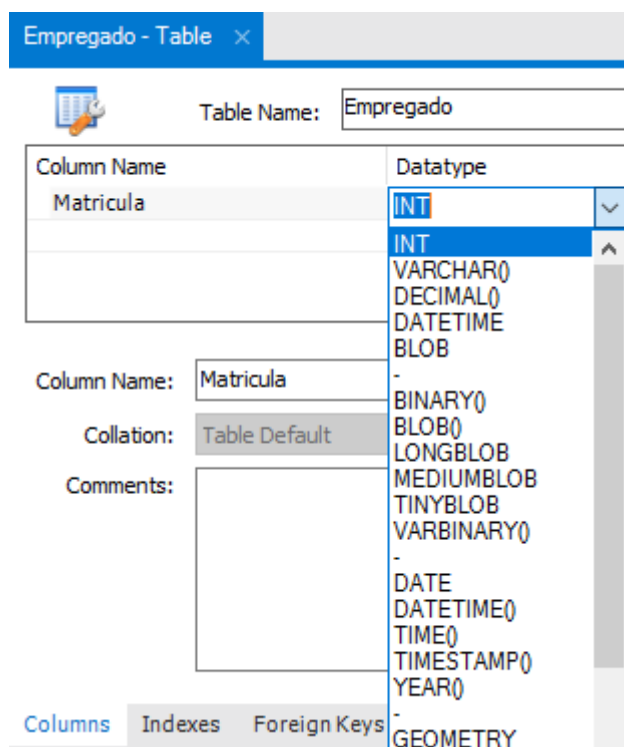
**Figura 09** - Exemplo de definição de nome de Atributos no *MySQL Workbench*.



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.

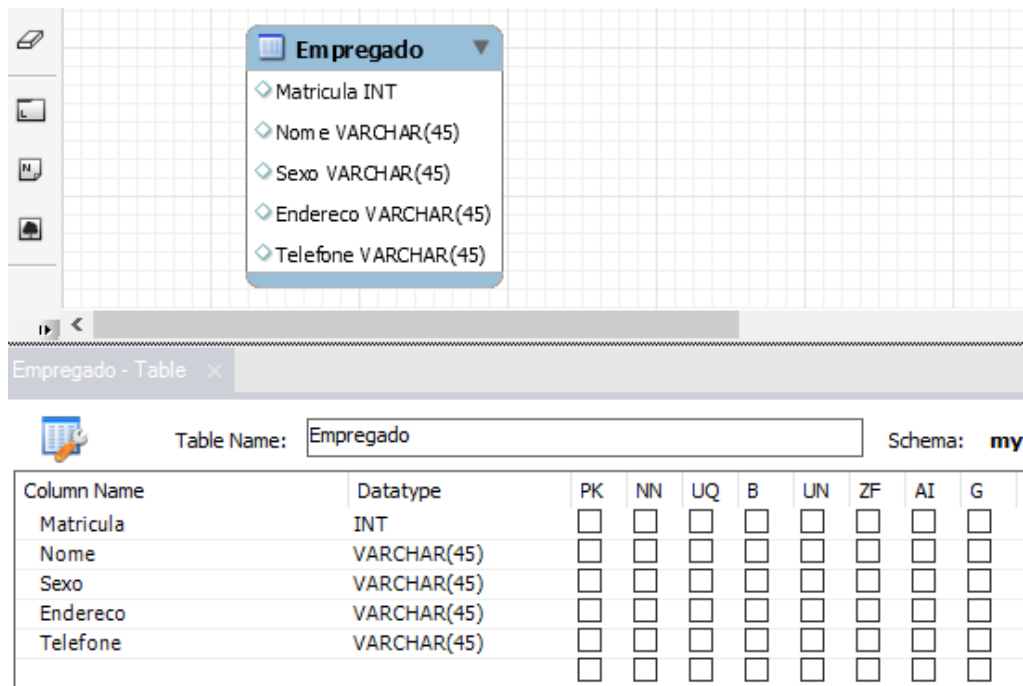
Depois de definir o nome dos atributos, você precisa definir os tipos de cada um. Para definir o tipo de cada atributo, você deve clicar na coluna *Datatype* de cada atributo. Ao clicar, irá aparecer uma lista com várias opções de tipos suportadas pelo *MySQL*, como mostrado pela **Figura 10**.

**Figura 10** - 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 11**.

**Figura 11** - Exemplo de definição de uma tabela no *MySQL Workbench*.



Note que os tipos dos atributos suportados variam entre os *SGBD*.

No caso do *MySQL*, veja no **Quadro 1** os principais tipos suportados são:

TIPO	DESCRIÇÃO
<b>VARCHAR</b>	Valores no campo VARCHAR são <i>strings</i> 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.
<b>INT</b>	Valores inteiros de -2147483648 a 2147483647.
<b>DECIMAL</b>	O tipo DECIMAL é usado por valores para os quais é importante preservar a exatidão como, por exemplo, dados monetários.
<b>DATE</b>	O tipo DATE é usado quando se necessita apenas do valor da data, sem a parte da hora. MySQL recupera e mostra valores do tipo DATE no formato 'ano-mm-dd'

**Quadro 1** - Descrição dos tipos de dados suportada pelo MySQL.

**Fonte:** Manual de Referência do MySQL 4.1.



### Vídeo 04 - Utilizando o MySQL

## Outros conceitos do modelo relacional

Um dos grandes desafios em se projetar um banco de dados com sucesso é a correta determinação das Tabelas que existirão nele, bem como dos atributos de cada tabela (BATISTI, 2010).

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

### Chave primária

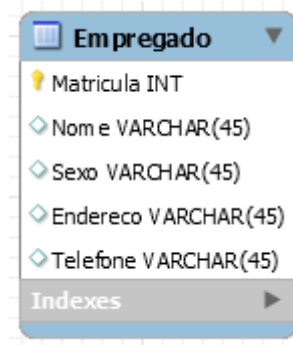
Entidades podem possuir um atributo chave que é usado para garantir a distinção entre as ocorrências das entidades. No Modelo Relacional, o conceito de chave primária equivale ao conceito de atributo chave do modelo ER. Um atributo que for chave primária em uma tabela não pode ter tuplas (registro) com 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 12, temos uma tabela com uma chave primária no atributo Matrícula.

**Figura 12** - Tabela com chave primária.

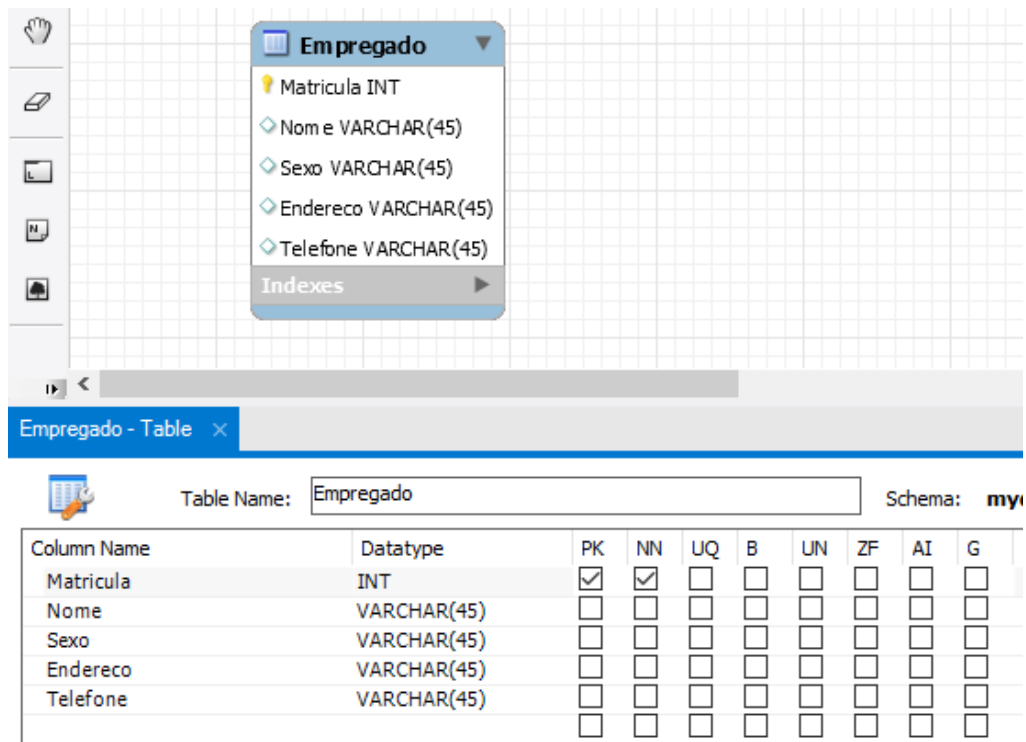


Bom, até agora aprendemos o que é uma tabela e seus atributos, mas, como vamos inserir uma chave primária?

Para inserirmos a chave primária, temos que marcá-la como tal. Na Figura 13 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*, e em português quer dizer chave primária.



**Figura 13** - Tabela com chave primária.



Empregado

- Matricula INT
- Nome VARCHAR(45)
- Sexo VARCHAR(45)
- Endereco VARCHAR(45)
- Telefone VARCHAR(45)
- Indexes

Empregado - Table

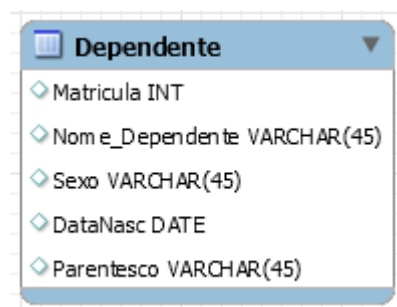
Table Name: Empregado Schema: myi

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
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"/>	<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"/>	<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"/>	<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"/>	<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"/>	<input type="checkbox"/>

## Chave estrangeira

Uma questão 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 14 mostra a tabela “Dependente”, que possui o atributo Matrícula. 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 14** - Tabela Dependente sem relação de integridade com a Tabela Empregado.



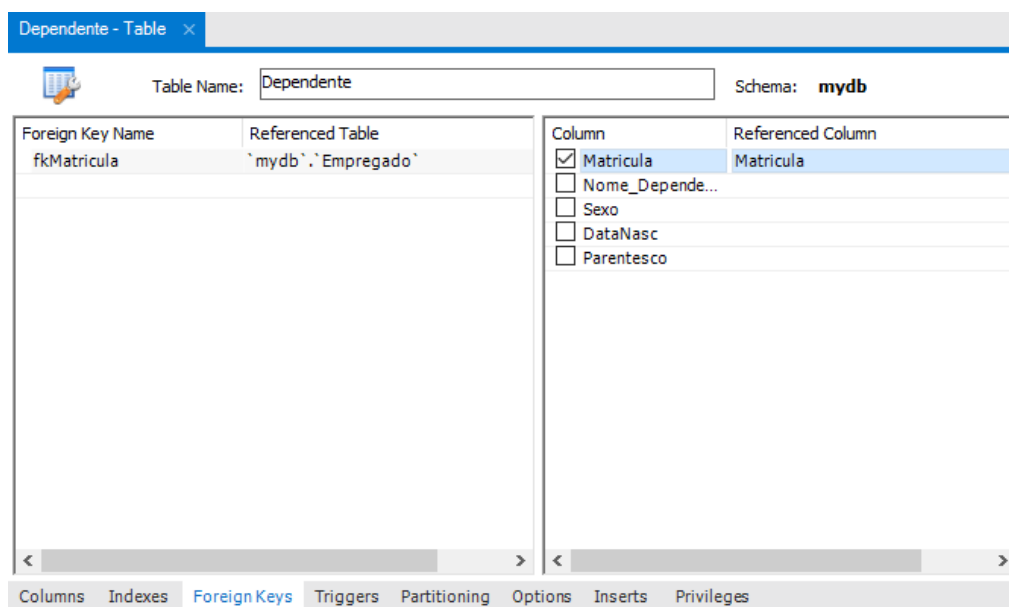
Dependente

- Matricula INT
- Nome\_Dependente VARCHAR(45)
- Sexo VARCHAR(45)
- DataNasc DATE
- Parentesco VARCHAR(45)

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”.

Bom, vamos aprender agora como inserimos nossa chave estrangeira utilizando a ferramenta MySQL Workbench. Para inserirmos a chave estrangeira, temos que fazer duas coisas. O primeiro passo é marcar nos quadradinhos ao lado do atributo Matrícula 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 15.

**Figura 15** - Definindo chave estrangeira no MySQL Workbench.



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 Matrícula.

Depois do exemplo da Figura 15, 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.

## Linguagem SQL

---

Todo **Sistema Gerenciador de Banco de Dados (SGBD)** deve oferecer aos seus usuários e administradores meios de criar definições de dados, bem como de manipular esses dados armazenados em suas bases. Desse modo, a linguagem SQL, sigla em inglês para *Structured Query Language*, ou Linguagem de Consulta Estruturada, é uma espécie de padrão adotado por diferentes SGBDs, de modo a facilitar a comunicação e a integração dos sistemas. Essa linguagem tem o objetivo de acessar, de forma estruturada e declarativa, elementos de tabelas de bancos de dados.

Além disso, é uma linguagem de pesquisa para uso em Banco de Dados Relacional. Você deve lembrar que estudou a linguagem Visualg, como exemplo de linguagem de programação estruturada, que permitiu a construção dos seus algoritmos, ou Java, como exemplo de linguagem orientada a objetos. Agora, vamos utilizar SQL como linguagem de construção, modificação e acesso a Banco de Dados. Cabe ressaltar que essa linguagem tem um caráter diferenciado das linguagens de programação, pois é uma linguagem declarativa, ou seja, não é necessário que você programe qualquer algoritmo para acessar o conteúdo das tabelas. Muito bom, não acha?

A linguagem SQL é um grande padrão de banco de dados, resultado da sua simplicidade e facilidade de uso. Ela se diferencia de outras linguagens de consulta a banco de dados, pois uma consulta SQL especifica a forma do resultado e não o caminho para chegar a ele.

Você pode se perguntar: qual a vantagem de existir uma linguagem padrão de acesso a banco de dados? Imagine se você estivesse numa reunião com um italiano, um chinês e um russo e ninguém falasse outra língua a não ser a de origem. Como se daria a comunicação? Provavelmente por mímicas e/ou desenho. Os sistemas computacionais ainda não conseguem utilizar esse tipo de estratégia. Desse modo, foi instituído um **padrão de comunicação** para Banco de Dados Relacional.

A linguagem SQL foi desenvolvida originalmente no início dos anos 1970 nos laboratórios da IBM em *San Jose*, dentro do projeto *System R*. O nome original da linguagem era **SEQUEL**, acrônimo para *Structured English Query Language* (Linguagem de Consulta Estruturada em Inglês). Embora a linguagem SQL tenha sido originalmente criada pela IBM, rapidamente surgiram vários "dialetos" produzidos por outros desenvolvedores. Essa expansão levou à necessidade de criar e de adaptar um padrão para a linguagem. Essa tarefa foi realizada pela *American National Standards Institute* (ANSI), em 1986, e pela *International Organization for Standardization* (ISO), em 1987. O SQL foi revisto três vezes, gerando a versão SQL:2003.

Embora SQL seja a linguagem de banco de dados mais influente do mercado, ela não é a única. A seguir, podemos conferir alguns exemplos de outras linguagens.

- **QBE** (*Query-by-Example*) – Integra o sistema de banco de dados QBE, foi desenvolvida pela IBM no início de 1970.
- **Quel** – Linguagem de consulta lançada para o banco de dados Ingres, desenvolvida na Universidade da Califórnia em *Berkeley*.
- **Datalog** – Linguagem de consulta baseada na lógica de programação Prolog.

SQL é caracterizada pela utilização de palavras-chaves, que podem ser classificadas, de acordo com sua função, nos seguintes tipos.

- **DML** – Linguagem de Manipulação de Dados, subconjunto da linguagem SQL, usado para inserir, atualizar e apagar dados. Exemplos: INSERT, UPDATE, DELETE.
- **DDL** – Linguagem de Definição de Dados, permite ao utilizador definir tabelas novas e elementos associados. Exemplos: CREATE, DROP.
- **DCL** – Linguagem de Controle de Dados, controla quem tem acesso para ver ou manipular dados dentro do banco de dados. Exemplos: GRANT, REVOKE.
- **DTL** – Linguagem de Transação de Dados, usada para o controle de transações no banco de dados. Exemplo: START TRANSACTION.
- **DQL** – Linguagem de Consulta de Dados, permite ao usuário especificar uma consulta (query) como uma descrição do resultado desejado. Exemplo: SELECT.

Todos esses termos estranhos, escritos em letra maiúscula, citados como exemplo, fazem parte da linguagem SQL. Nas próximas aulas, você descobrirá para que eles servem e como utilizá-los.

Já utilizamos na aula de hoje, algumas vezes, o termo **consulta**, mas o que seria isso? Quando precisamos de uma informação que está armazenada em um banco de dados, dizemos que vamos consultá-lo. Imagine que você precisa ligar para um amigo, mas não lembra o número do telefone dele. O que você faz? Digita o nome dele na agenda do seu telefone, fazendo uma consulta.



## Vídeo 02 - Vantagens da Linguagem SQL

### Atividade 01

---

1. Descreva alguma situação do seu dia a dia em que você precise utilizar um banco de dados. Especifique um exemplo de consulta que você pode fazer por meio dele.

### Sistemas Gerenciadores de Banco de Dados

---

Os Sistemas Gerenciadores de Banco de Dados (SGBDs) são conjuntos de programas de computador (softwares) responsáveis pelo gerenciamento de um banco de dados. O principal objetivo é retirar da aplicação em desenvolvimento a responsabilidade de gerenciar o acesso, manipulação e organização dos dados. O SGBD disponibiliza uma interface para que os seus clientes possam incluir, alterar ou consultar dados.

Todas as organizações possuem grandes quantidades de dados e informações que precisam armazenar. Contudo, nem sempre o armazenamento de dados em papel é confiável e de fácil acesso. Nesse sentido, torna-se mais fácil encontrar a informação num banco de dados que se utilize das tecnologias de informação atuais para seu gerenciamento. Ou seja, os bancos de dados estendem a função do papel ao guardar a informação em computadores.

Qualquer empresa que pretenda garantir um controle efetivo sobre o seu negócio tem obrigatoriamente que recorrer a SGBDs. A planilha eletrônica é uma ferramenta de controle usada para operacionalizar os dados e assim criar informação útil ao planejamento diário das empresas. Contudo, existem outros tipos de ferramentas, mais completas e com funcionalidades extras que elevam a capacidade operacional de gerenciar informação de valor para a organização.

Um SGBD não é nada mais do que um conjunto de programas que permitem armazenar, modificar e extrair informação de um banco de dados. Há diferentes tipos de SGBD, desde pequenos sistemas que funcionam em computadores pessoais, até sistemas complexos associados a servidores de grandes empresas.



### **Vídeo 03** - Características de um SGBD

Os SGBDs mais populares que utilizam linguagem SQL estão listados a seguir.

- **Oracle** – surgiu no fim dos anos 70, desenvolvido pela empresa de mesmo nome, que é líder de mercado no segmento de banco de dados, sendo pioneira no lançamento de novas soluções de banco de dados para diversos segmentos, sobretudo, no campo das aplicações corporativas. Atualmente, encontra-se na versão *Oracle 12c*.
- **SQL Server** – criado pela *Microsoft* em 1989, surgiu de uma parceria entre a *Sybase*, *Ashton-Tate* e *Microsoft*. Atualmente, encontra-se na versão *Microsoft SQL Server 2017*, sendo considerada uma plataforma de dados confiável, produtiva e inteligente.
- **PostgreSQL** – SGBD de código aberto do tipo objeto relacional lançado em 1995, resultante de uma evolução do projeto Ingres, desenvolvido na Universidade da Califórnia em *Berkeley*. Encontra-se atualmente na versão 10.
- **Firebird** – SGBD de código aberto, criado a partir do *InterBase* da *Borland*, quando da abertura de seu código em julho de 2000. Encontra-se atualmente na versão 3.0.2.
- **MySQL** – SGBD de código aberto, criado na década de 1980 na Suécia, considerado o SGBD *open source* mais popular no mundo atualmente, sendo utilizado por grandes corporações, como NASA, Banco Bradesco, Dataprev, HP, *Nokia*, *Sony*, *Cisco Systems*, *Google* e outros. O MySQL foi comprado em 2008 pela

SUN *Microsystems*, que posteriormente foi comprada pela *Oracle*. Ou seja, o MySQL atualmente é um produto da *Oracle*. O seu sucesso é atribuído à sua fácil integração com PHP presente na maioria das ferramentas de desenvolvimento para Internet. Encontra-se atualmente na versão 5.7.



#### **Vídeo 04** - Arquitetura ANSI/SPARC

## Atividade 02

---

1. Descubra outras características de um dos SGBDs listados anteriormente, consultando sites de busca. Se a página acessada estiver em inglês, será uma ótima oportunidade de você treinar o que aprendeu na disciplina de inglês técnico.

Considerando o resumo apresentado na lista anterior, escolhemos o MySQL como exemplo de SGBD a ser utilizado em nossa disciplina. A partir de agora, vamos ensinar você a instalar e configurar o ambiente. Após isso, você já estará apto a entrar no mundo maravilhoso da linguagem SQL.

## Instalando o *MySQL*

---

Como vimos anteriormente, o MySQL é o SGBD de código aberto mais popular no mundo atualmente. Por isso, nós o utilizaremos nas nossas aulas. A versão que utilizaremos será a versão MySQL Community Server 5.7.21. Versões mais recentes deverão ser lançadas e possivelmente poderão ser compatíveis com nosso material.

Para instalar o MySQL Server Community será necessário realizarmos o download do MySQL Installer. O MySQL Installer é uma ferramenta criada para auxiliar no processo de instalação e configuração dos produtos MySQL. Para obter o

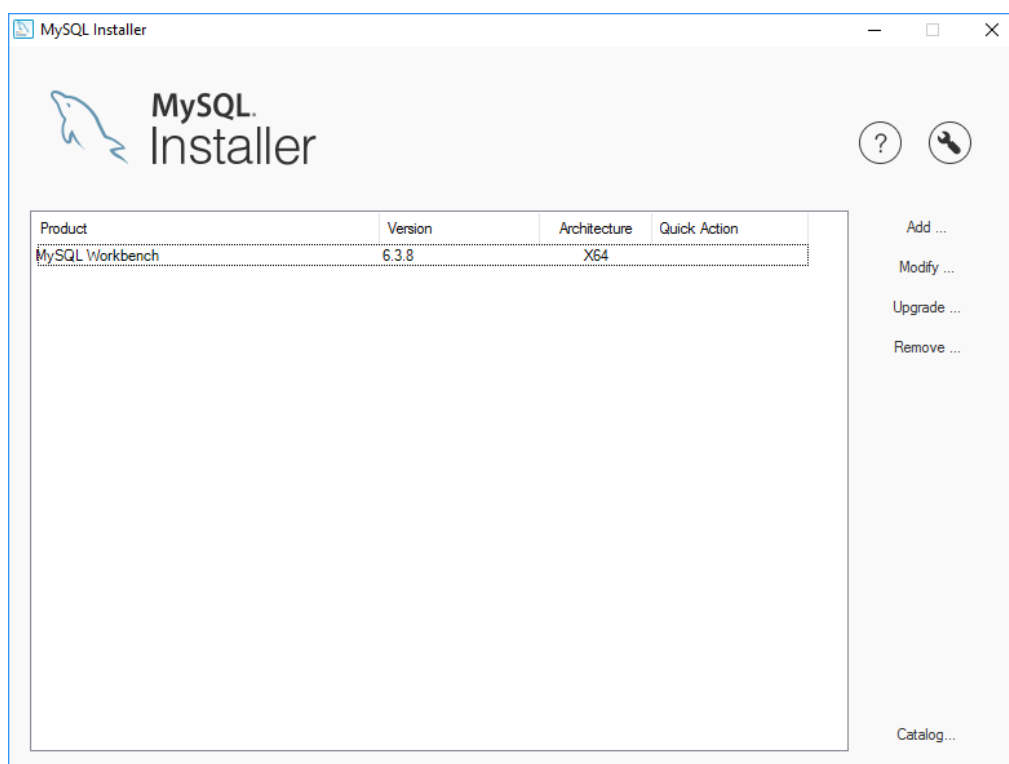


MySQL Installer 5.7 para Windows, você poderá acessar a página do link <<https://dev.mysql.com/downloads/mysql/>>. Após o carregamento, escolha a opção “Microsoft Windows” como sistema operacional e sua respectiva versão. Em seguida, localize a opção “MySQL Installer 5.7 for Windows” e clique em “Go to Download Page”.

Nesta página, existem duas opções do MySQL Installer: a primeira opção, com menos de 19MB, somente realizará o download dos arquivos do MySQL Server no momento da instalação. Já a segunda opção do MySQL Installer, com mais de 370MB, os arquivos necessários para a instalação do MySQL Server já estarão disponíveis no momento da instalação. Recomendamos o download dessa segunda opção.

Após baixar e executar o MySQL Installer, você verá a tela apresentada na Figura 16. Clique em “**Add ...**” para iniciarmos o processo de instalação do MySQL Server.

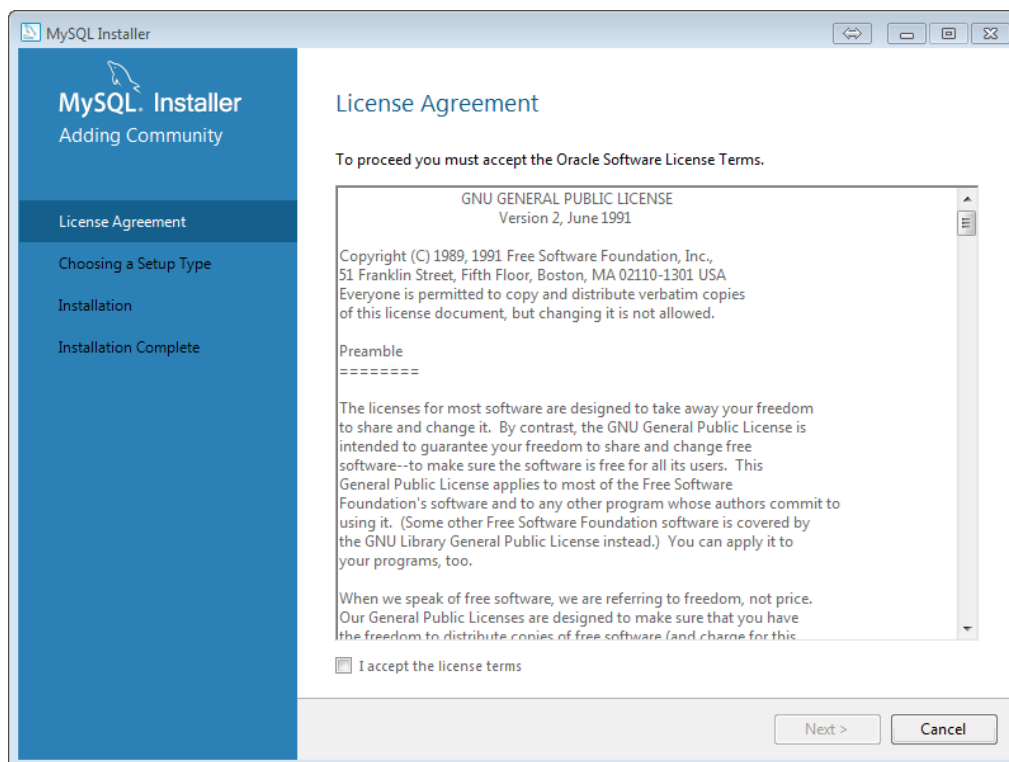
**Figura 16** - Tela Inicial do MySQL Installer.



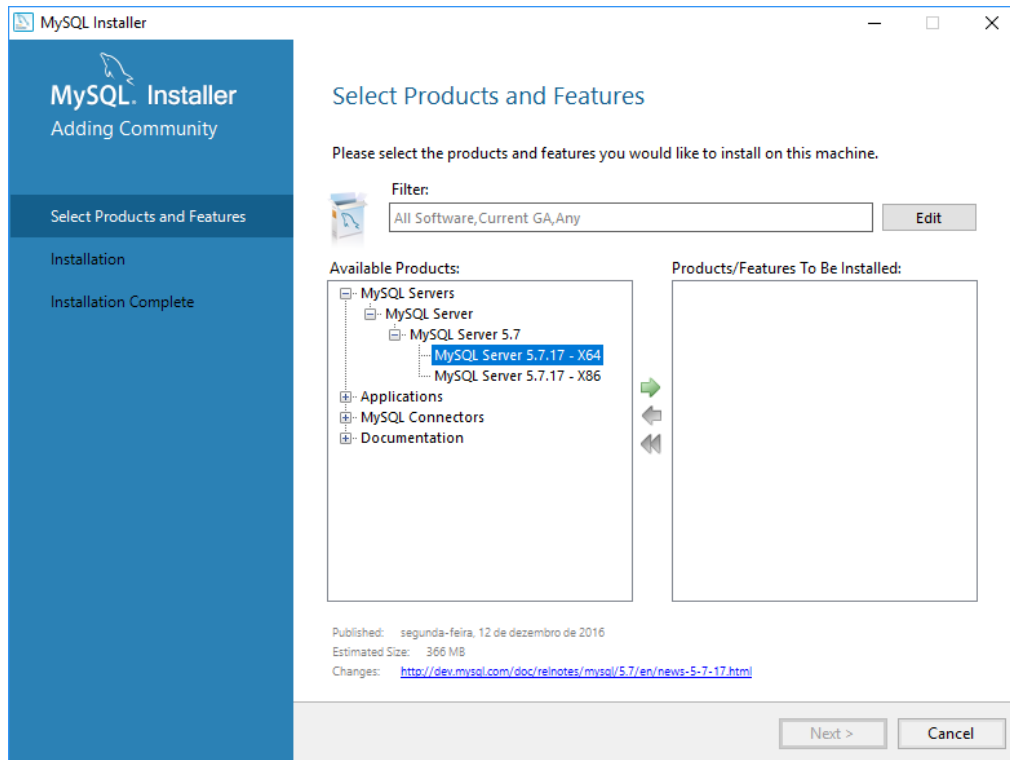
Em seguida, você verá a tela apresentada na Figura 17. Marque a caixa “I Accept the license terms” para aceitar os termos de licença do MySQL e em seguida clique em “Next” para prosseguir com a instalação.

Na tela da Figura 18 você deverá expandir a as opções referentes ao MySQL Server. Note que há duas versões disponíveis para a instalação: a versão para sistemas operacionais de 64bits e a versão de 32bits. Você deverá selecionar a versão que for adequada ao seu sistema operacional. Em nosso exemplo estamos realizando a instalação no Windows de 64bits.

**Figura 17** - Instalação do MySQL Server.

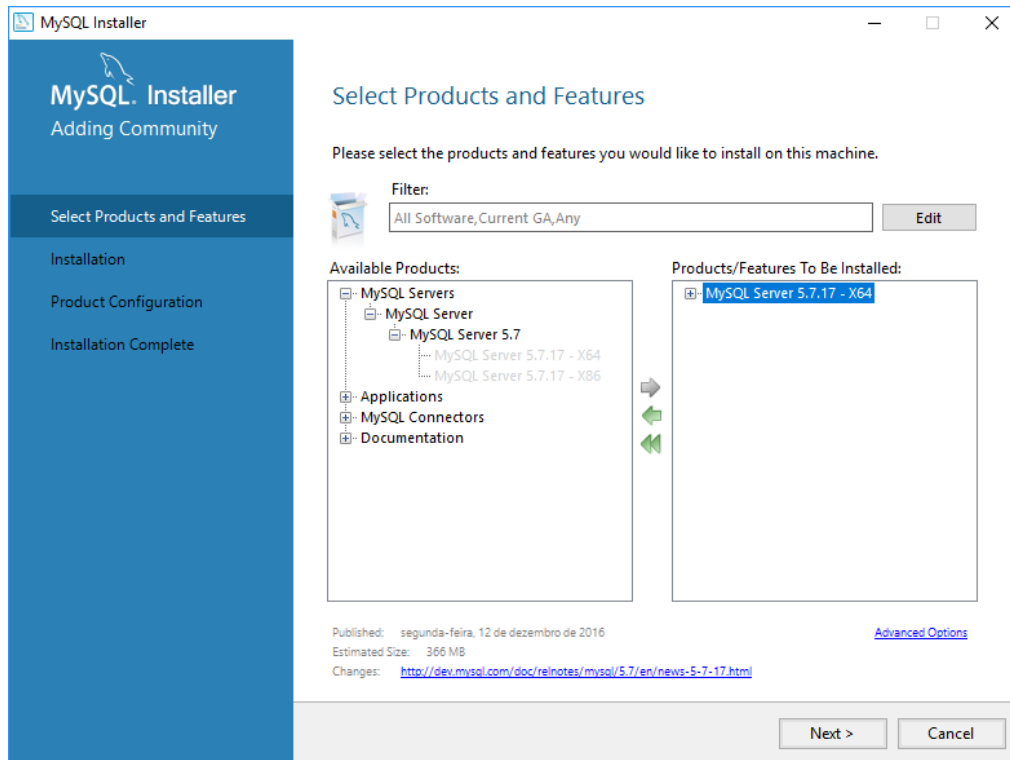


**Figura 18** - Instalação do MySQL Server.



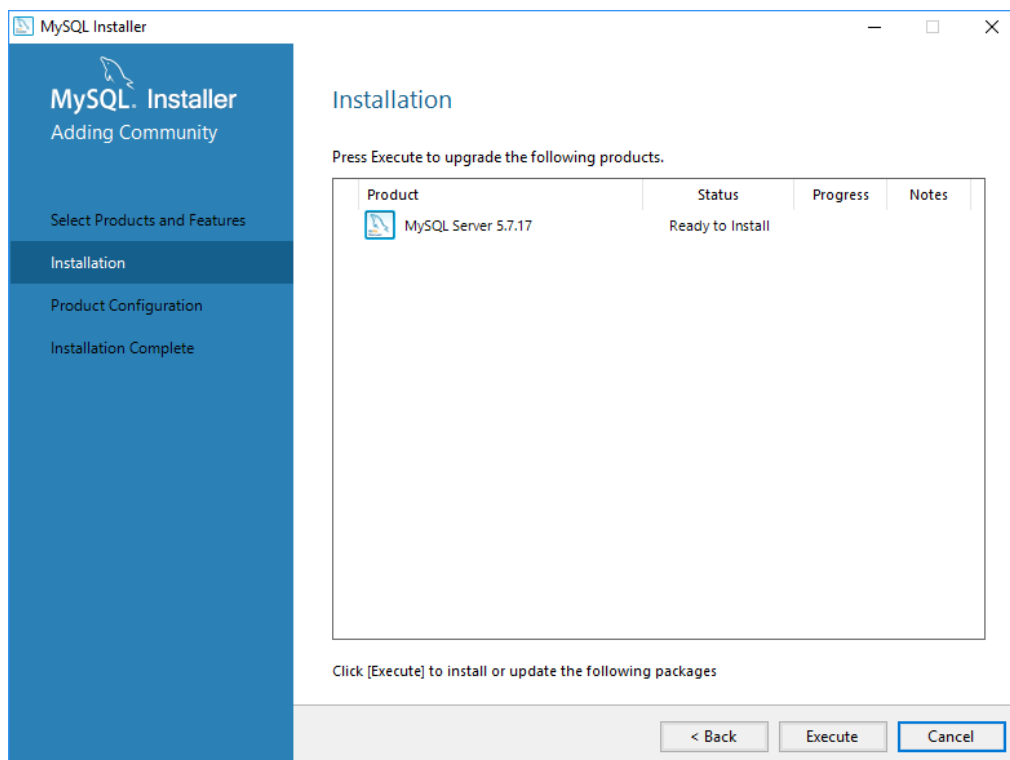
Para selecionar a versão que será instalada, clique sobre o “MySQL Server 5.7.17 – X64” e então clique na seta localizada ao centro da janela e que está apontando para a caixa a direita. Após clicar a seta a tela deverá ser semelhante ao apresentado na Figura 19. Clique em *Next >*.

**Figura 19** - Instalação do MySQL Server.



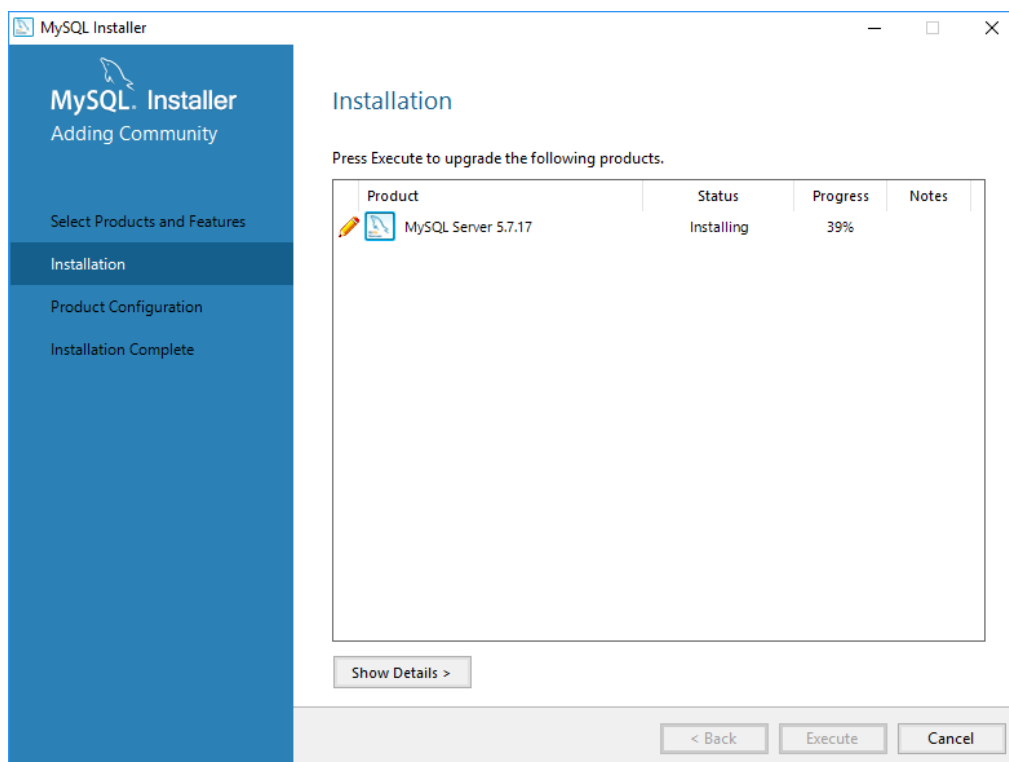
A tela seguinte, Figura 20, é apresentado um resumo do que será instalado. Clique então em *Execute* para continuar o processo de instalação.

**Figura 20** - Instalação do MySQL Server.

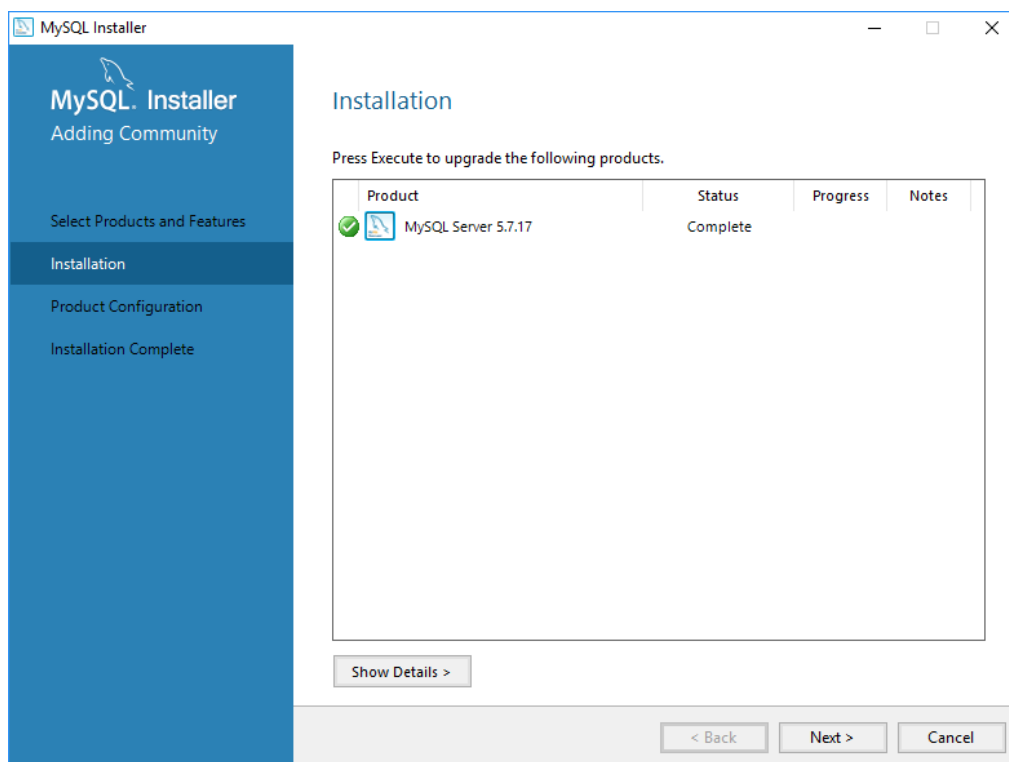


A Figura 21 mostra o progresso da instalação do MySQL Server, já a Figura 22 indica que o processo de instalação foi concluído. Clique no botão *Next >*.

**Figura 21** - Instalação do MySQL Server.

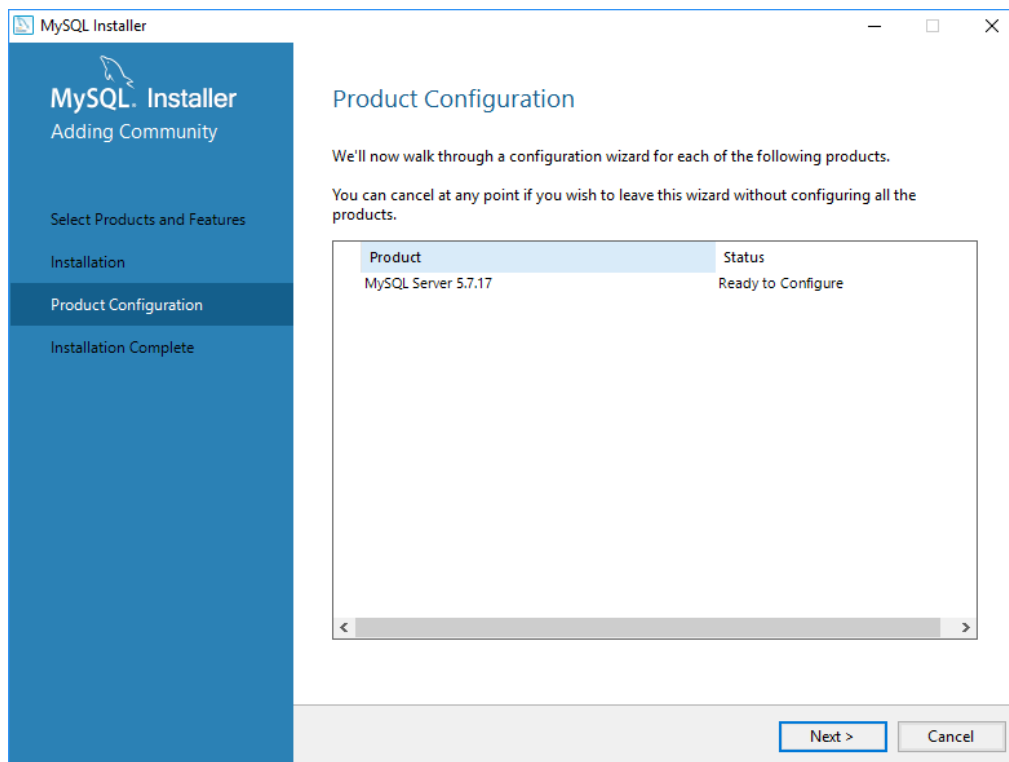


**Figura 22** - Instalação do MySQL Server.



Após a instalação do MySQL Server será apresentada a tela da Figura 23. A partir desse momento iremos realizar a configuração inicial do serviço do banco de dados.

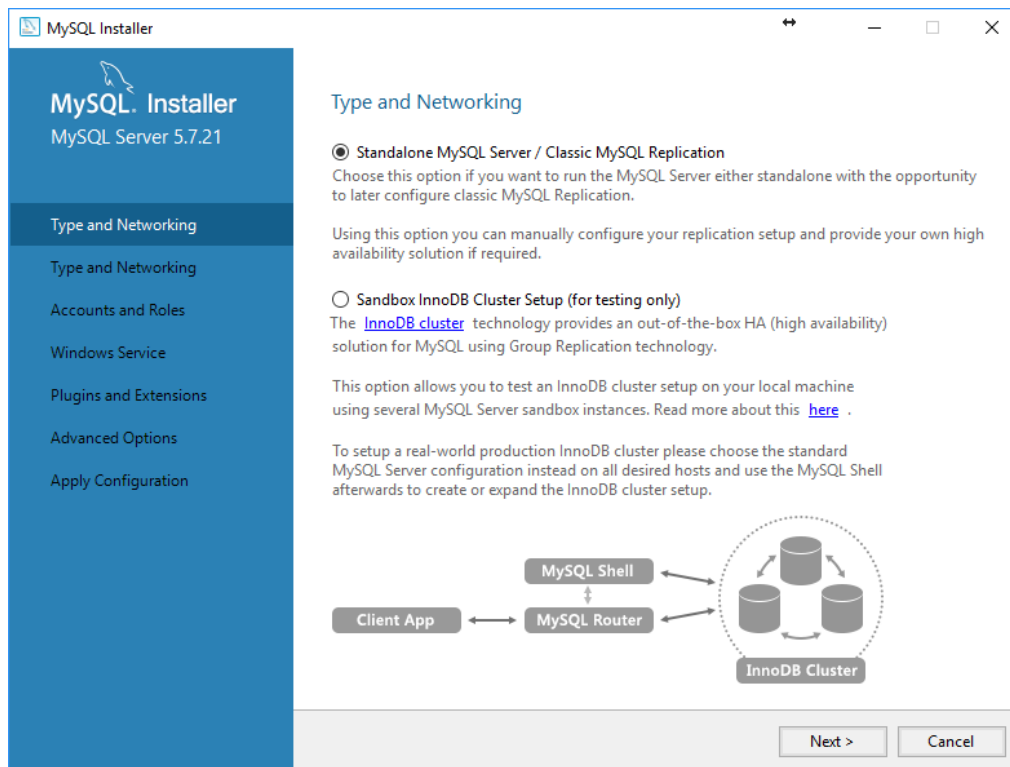
**Figura 23** - Configuração do MySQL Server após a instalação.



Ao iniciar a configuração você será apresentado a tela como mostra a Figura 24. Selecione a opção "Standalone MySQL Server/Classic MySQL Replication". Nesta opção o MySQL Server será instalado no modo *standalone*, ou seja, não utilizará recursos de cluster. Para o propósito de nosso curso é o suficiente. Clique em "Next" para prosseguir com a configuração.

A título de curiosidade, a opção "Sandbox InnoDB Cluster Setup" auxilia a configuração do MySQL Server em uma arquitetura de cluster, ou seja, quando se utiliza múltiplas instâncias para servir o banco de dados. Neste modo, além de outras vantagens, o sistema funciona em alta disponibilidade, ficando mais tolerante a falhas de um ou mais instâncias. Trata-se de uma arquitetura mais complexa utilizada para sistemas em produção que possuam requisitos de alta disponibilidade.

**Figura 24** - Instalação do MySQL Server.

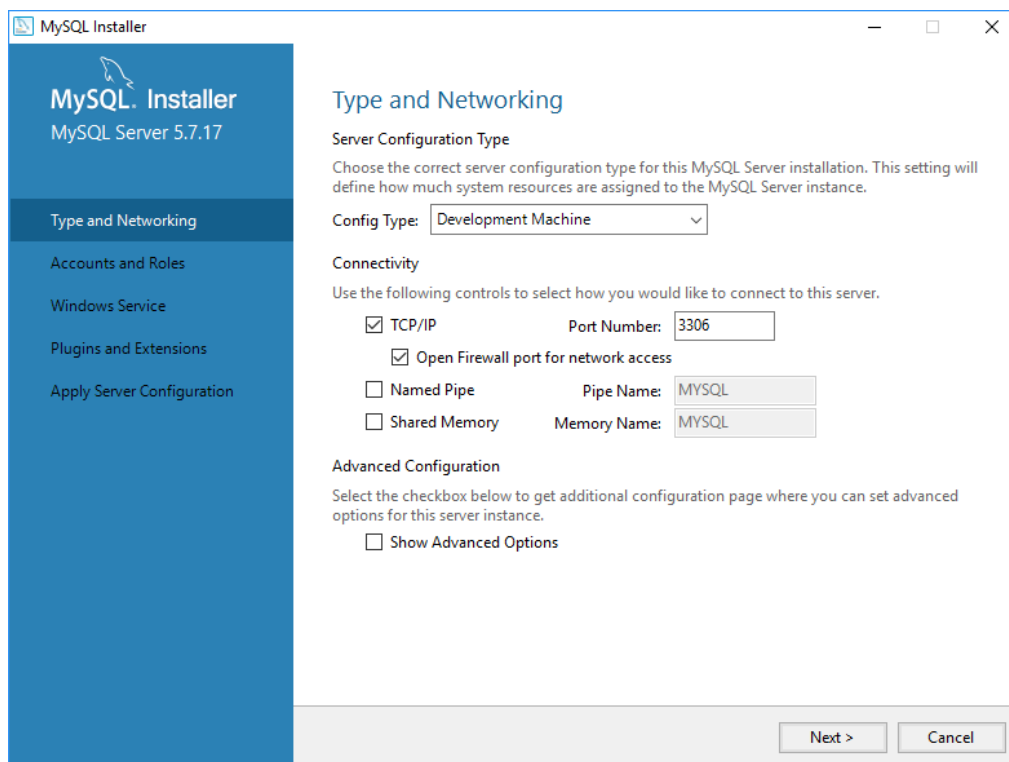


A Figura 25 apresenta o tipo de configuração que desejamos em Config Type. No nosso caso vamos utilizar a opção “Development Machine”, já que a nossa instalação será utilizada para o desenvolvimento de nossas atividades e não necessitará do desempenho máximo do MySQL Server.

As outras opções disponíveis são utilizadas quando vamos montar um “Ambiente de Produção” para a execução de Softwares e Aplicações que necessitam de todo o desempenho disponível da máquina em que o banco de dados está instalado.

Ainda na tela apresentada na Figura 25 é possível definir o número da porta que o MySQL Server deverá aceitar as conexões e a porta padrão é a 3306. Defina suas opções conforme o que está apresentado e então clique em *Next >*.

**Figura 25** - Configuração do MySQL Server após a instalação.



Na tela seguinte, Figura 26, devemos definir a senha do usuário “root”. Esse usuário possui a permissão para criar, apagar ou modificar relações/tabelas, registros e os usuários cadastrados no banco de dados.

A senha para esse usuário não poderá possuir menos de 4 caracteres. Como estamos instalando um ambiente de desenvolvimento devemos definir uma senha fácil de ser memorizada. Podemos utilizar a palavra “senha” (sem aspas e toda e minúsculas) nessa instalação. Após inserir a senha nas duas caixas de texto, clique em *Next >*.



**Figura 26** - Configuração do MySQL Server após a instalação.

MySQL Installer  
MySQL Server 5.7.17

Type and Networking  
Accounts and Roles  
Windows Service  
Plugins and Extensions  
Apply Server Configuration

### Accounts and Roles

**Root Account Password**  
Enter the password for the root account. Please remember to store this password in a secure place.

MySQL Root Password:

Repeat Password:

Password minimum length: 4

**MySQL User Accounts**  
Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

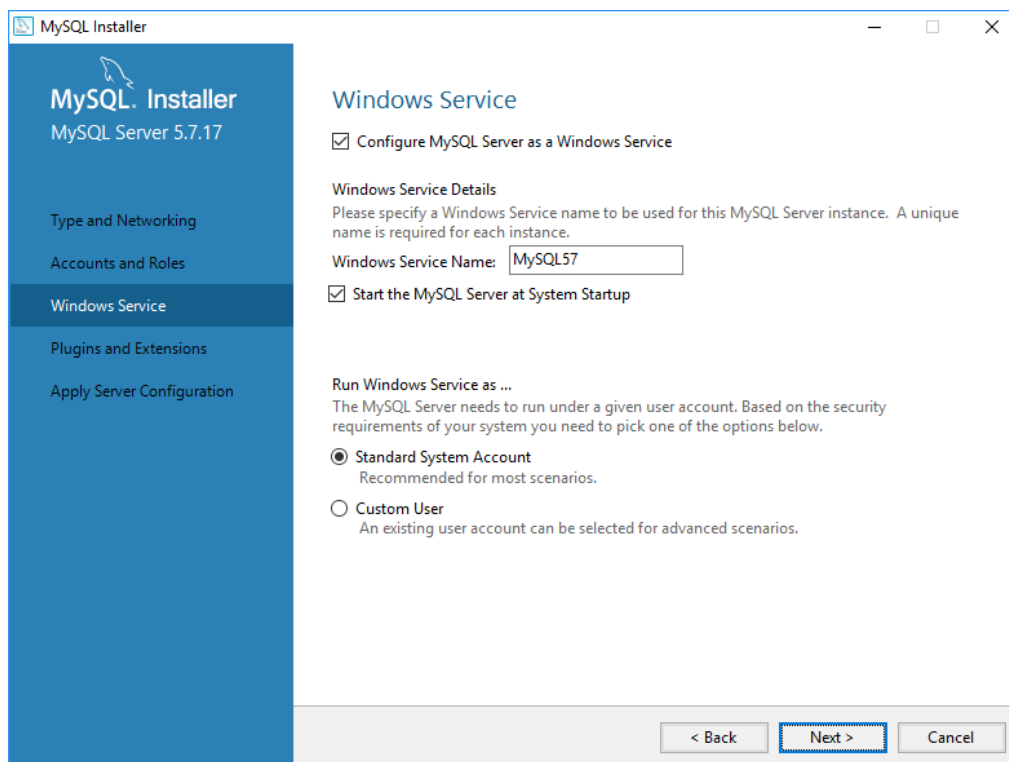
MySQL Username	Host	User Role
----------------	------	-----------

Add User  
Edit User  
Delete

< Back   Next >   Cancel

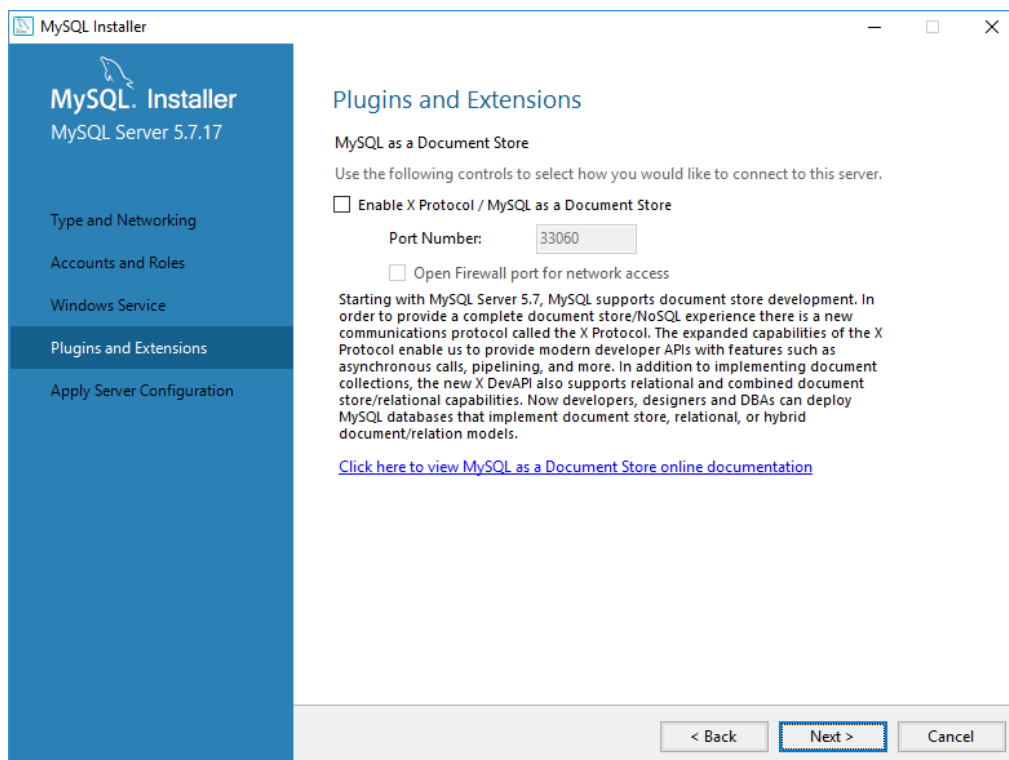
Na tela da Figura 27 podemos definir se o MySQL Server será executado como um serviço, o nome do serviço (MySQL57) e se ele deverá ser inicializado junto com o sistema operacional. Não iremos realizar modificações nessa etapa, clique em *Next >*.

**Figura 27** - Configuração do MySQL Server após a instalação.



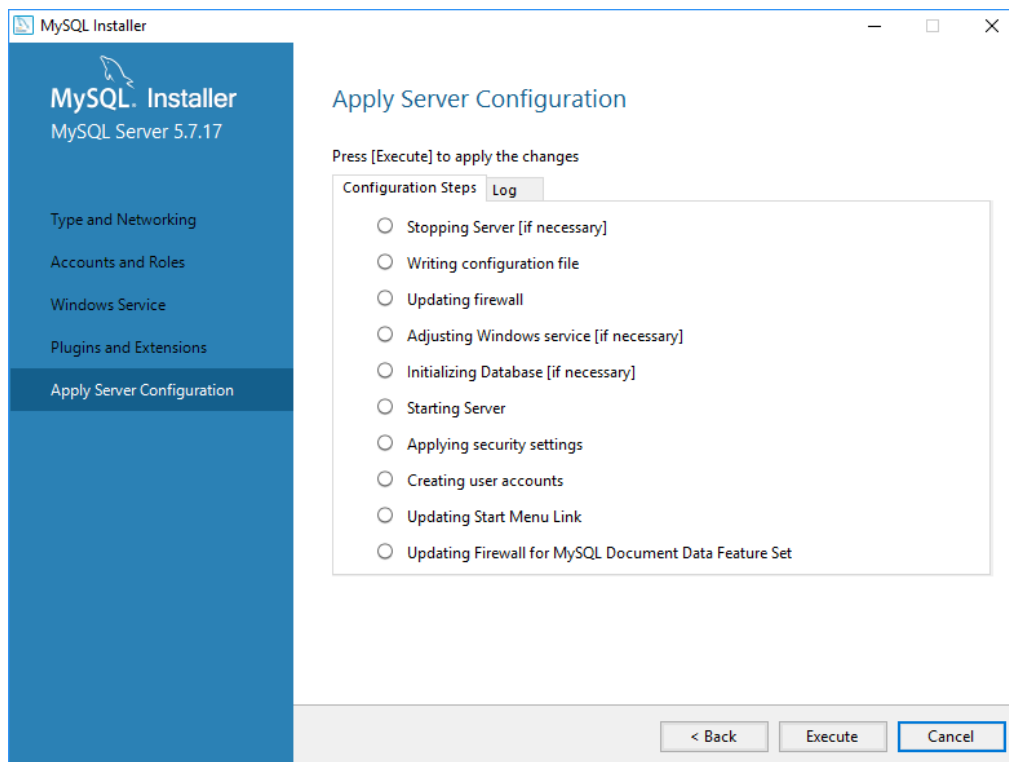
Na tela da Figura 28 podemos ativar o protocolo adicional. Não vamos necessitar realizar nenhuma configuração nessa tela, devemos então clicar em *Next >*.

**Figura 28** - Configuração do MySQL Server após a instalação.



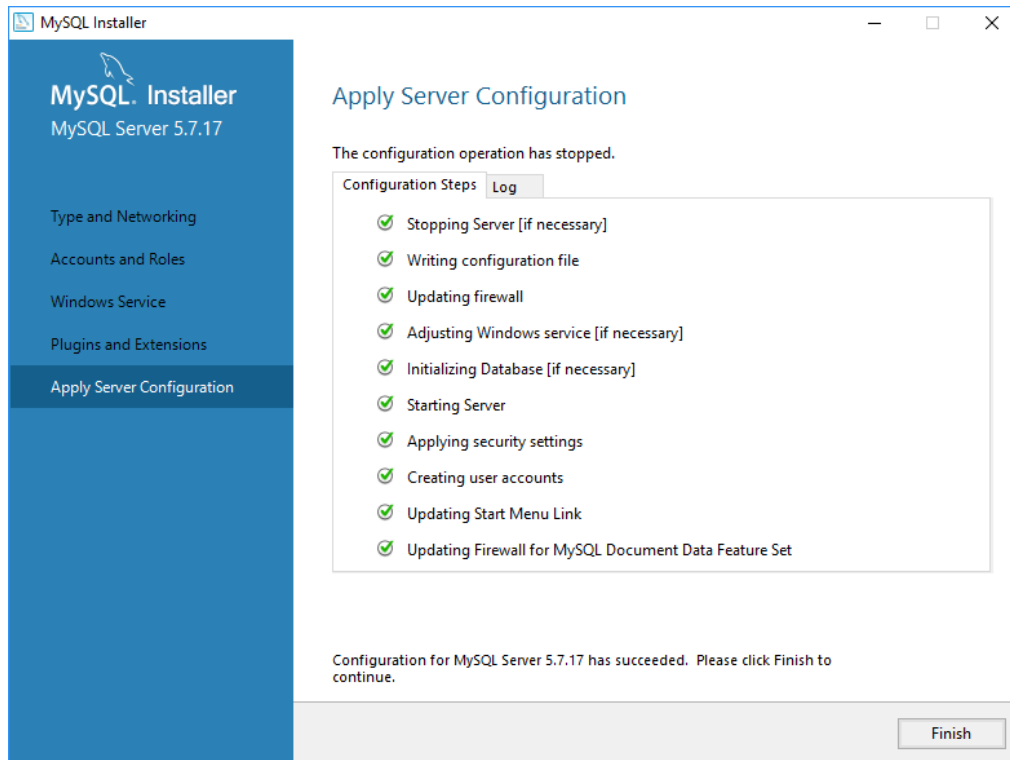
Na próxima tela, Figura 29, são apresentados os passos da configuração que realizamos. Para que possamos aplicar as configurações definidas até o momento devemos clicar no botão *Execute*.

**Figura 29** - Configuração do MySQL Server após a instalação.

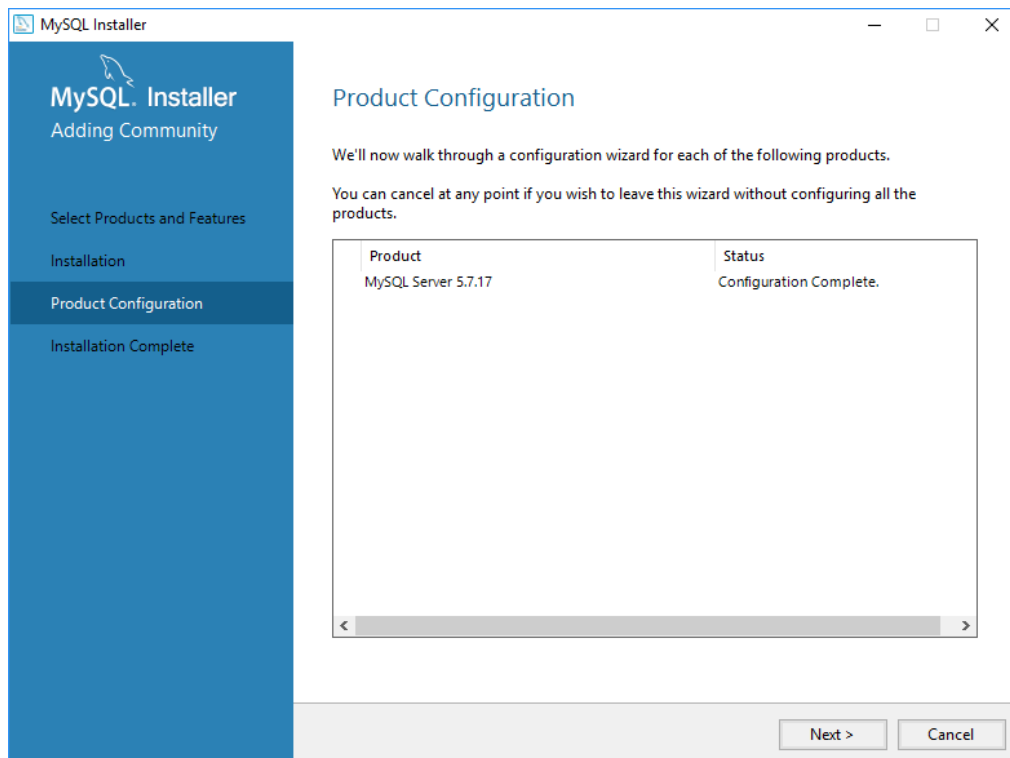


A tela da Figura 30 apresenta que todos os passos foram concluídos com sucesso. Clique em *Finish* para finalizar o processo de instalação e configuração do MySQL Server.

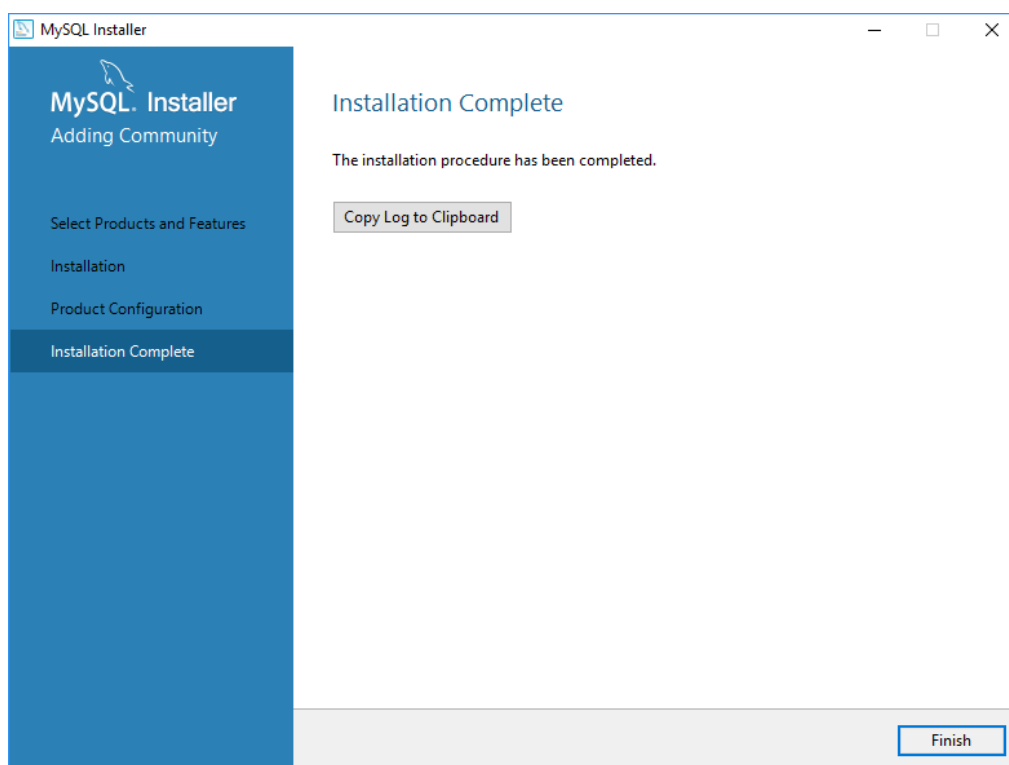
**Figura 30** - Configuração do MySQL Server após a instalação.



**Figura 31** - Configuração do MySQL Server após a instalação.



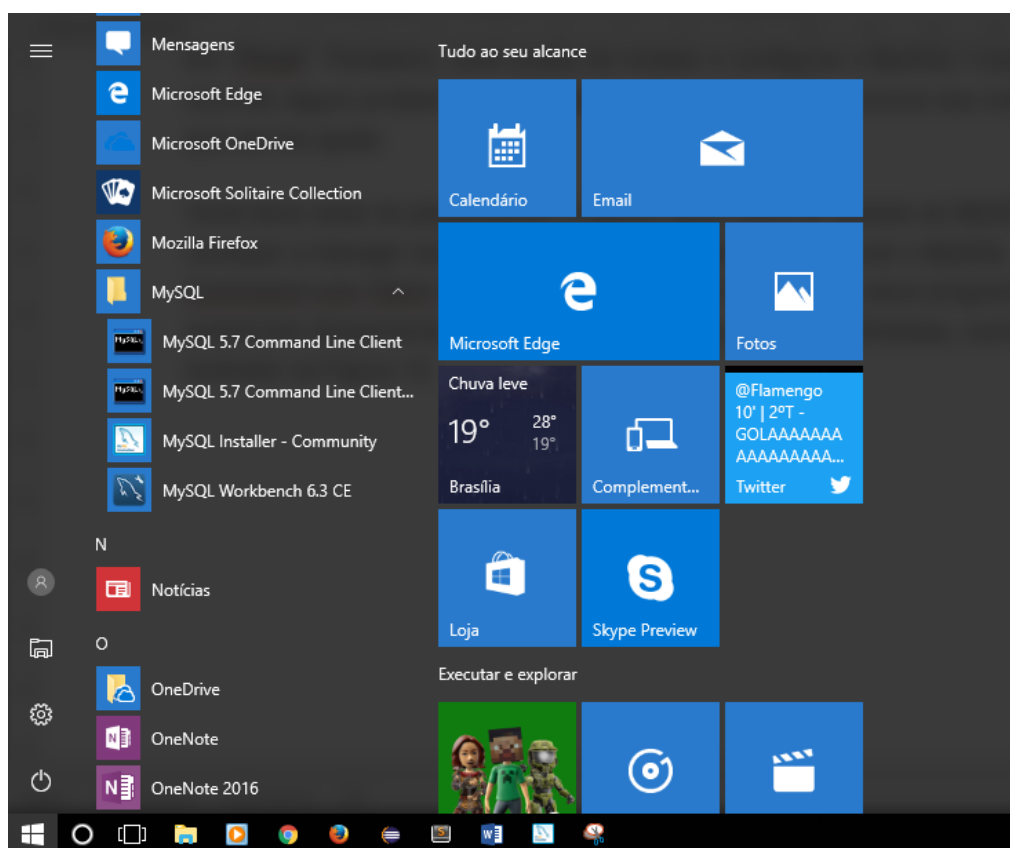
**Figura 32** - Configuração do MySQL Server após a instalação.



Parabéns, você acaba de instalar e configurar o MySQL! Caso tenha ocorrido algum problema na instalação e/ou configuração, procure seu tutor para que ele lhe ajude.

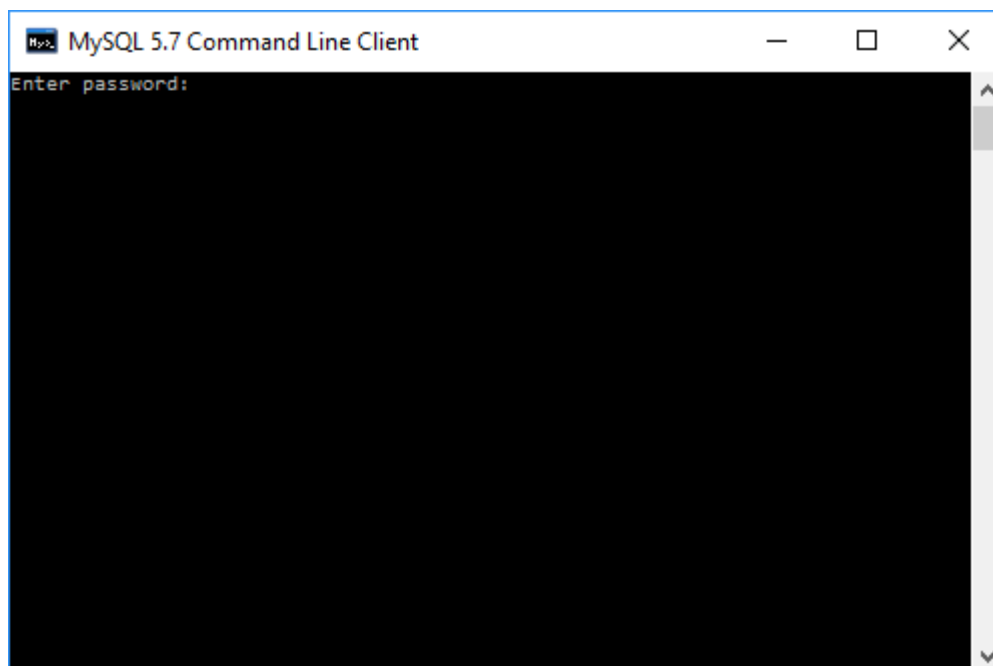
Você deve estar se perguntando: e agora? Bom, para ter acesso ao MySQL e começar a interagir com seu banco de dados, você irá procurar o *MySQL Command Line Client* no menu MySQL dentro dos seus programas, acessível diretamente pelo menu Iniciar do seu ambiente Windows, como ilustrado na Figura 33.

**Figura 33** - Acesso ao MySQL Command Line Client pelo ambiente Windows.



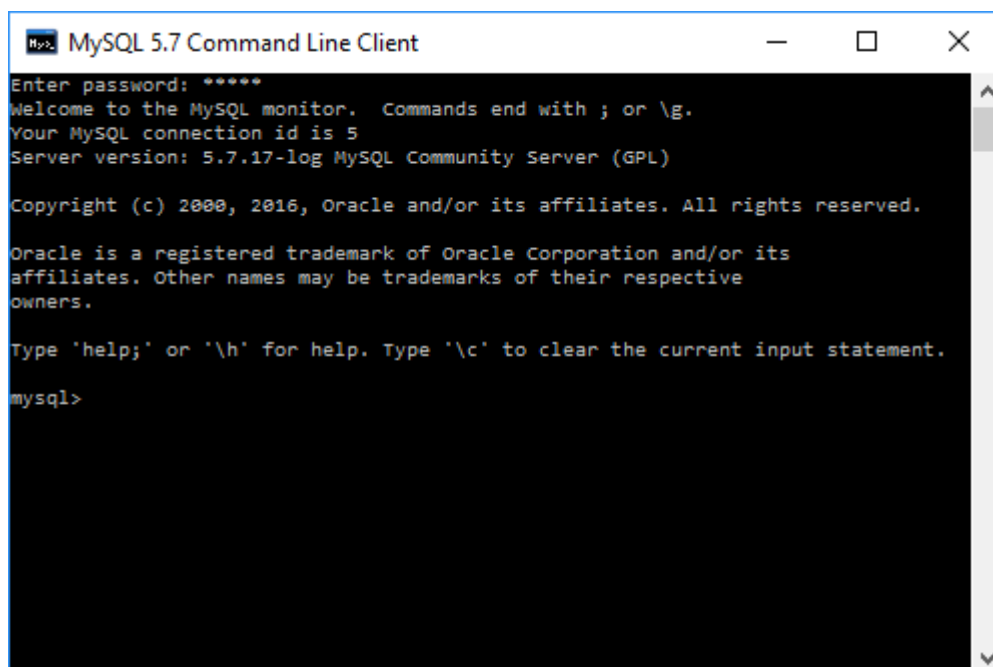
Ao clicar no ícone, aparecerá a tela de entrada do MySQL (Figura 34).

**Figura 34**



Na tela inicial do MySQL, é solicitado a você que digite a senha de root que foi configurada durante a configuração do ambiente. Digite a senha e tecele enter. Uma nova tela aparecerá dando-lhe as boas-vindas ao ambiente MySQL (Figura 35).

Figura 35



Agora, já podemos começar a interagir com a linguagem SQL. Que tal criar seu primeiro banco de dados? Digite a sequência de comandos apresentada no quadro a seguir. Não esqueça de teclar *enter* após o sinal de ponto e vírgula.

1	mysql> CREATE DATABASE metropole;
2	mysql> SHOW DATABASES;
3	

O primeiro comando (CREATE DATABASE) é usado para criar bancos de dados. No exemplo, criamos um banco de dados intitulado de metropole. O segundo comando (SHOW DATABASES) é usado para mostrar os bancos de dados existentes no servidor. Você viu que, além do banco de dados que você criou, existem outros denominados de *information\_schema*, *performance\_schema*, *sys* e *mysql*. Eles foram criados automaticamente durante a instalação do programa.

Tente executar os comandos acima, escrevendo os comandos SQL de forma errada, por exemplo, SHOW DATABASE, ou sem digitar o ponto e vírgula no final. Veja o que acontece.

Se você tiver alguma dúvida sobre a escrita ou a função de algum comando, pode digitar *"help nome\_do\_comando"* e teclar *enter* que você terá uma breve descrição do comando. Tudo certo? Pegue sua agenda e anote ao lado da data de hoje: "criei meu primeiro banco de dados em SQL!".



**Vídeo 05** - Instalando o MySQL

## Atividade 03

---

1. Você deve ter percebido que o acesso ao MySQL é feito diretamente pela linha de comando. Será que existe alguma interface gráfica que facilite essa interação? Faça uma pesquisa na Internet e verifique se existe alguma ferramenta visual para acesso ao MySQL.



# Resumo

---

Nesta aula, você verificou que existe uma linguagem padrão de acesso aos bancos de dados, denominada de SQL, e conheceu um pouco de sua história. Em seguida, aprendeu que, para usar um banco de dados, é necessário instalar um Sistema Gerenciador de Banco de Dados (SGBD) e que existem diversas opções com características variadas. Aprendeu, também, a instalar e configurar a ferramenta MySQL. Agora que você já tem o SGBD instalado, você está pronto para explorar a linguagem SQL em toda a sua profundidade. Será isso que faremos a partir da próxima aula.

## Autoavaliação

---

1. O que é e para que serve a SQL?
2. SQL é a única linguagem utilizada em banco de dados? Justifique.
3. O que é uma consulta em banco de dados?
4. O que é e para que serve um SGBD?
5. Cite duas vantagens existentes na escolha do MySQL como SGBD para nossas aulas.

## Referências

---

BEIGHLEY, L. **Use a cabeça SQL**. Rio de Janeiro: Editora AltaBooks, 2008.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. **Sistema de banco de dados** 8. 5. ed. Rio de Janeiro: Editora Campus/Elsevier, 2006.